OIPE JC167 JAN 1 2 2004 PATENT & TRADEMARK OFFICE

#7/Dec
24065/50
1-2004

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:

Chijioke Chukwuemeka

Examiner: Naeem U. Haq

Group Art Unit: 3625

Serial No.: 09/650,293

Filed: August 29, 2000

For: METHOD AND APPARATUS FOR MAKING SECURE ELECTRONIC PAYMENTS

Commissioner of Patents
P.O. Box 1450
Alexandria, VA 2233-1450

RECEIVED
JAN 1 5 2004
GROUP 3600

DECLARATION UNDER 37 C.F.R. § 1.131

S I R:

Chijioke Chukwuemeka, declares as follows:

1. I am the sole inventor of the invention in the above-referenced patent application and all the subject matter described and claimed therein. I submit this Declaration in order to establish a date of invention prior to the earliest priority filing date of January 26, 2000 of U.S. Patent Application Publication No. 2000/0002538 (the "Ling Publication"), a reference cited during the examination of the patent application.

2. Prior to January 26, 2000, I conceived of the invention as described in the claims set forth in the patent application, and with due diligence from prior to January 26, 2000 to the filing of this application, I undertook efforts to reduce the invention to practice. Annexed hereto as Exhibit A are copies of various computer-printed documents which were generated prior to January 26, 2000. The cover page of

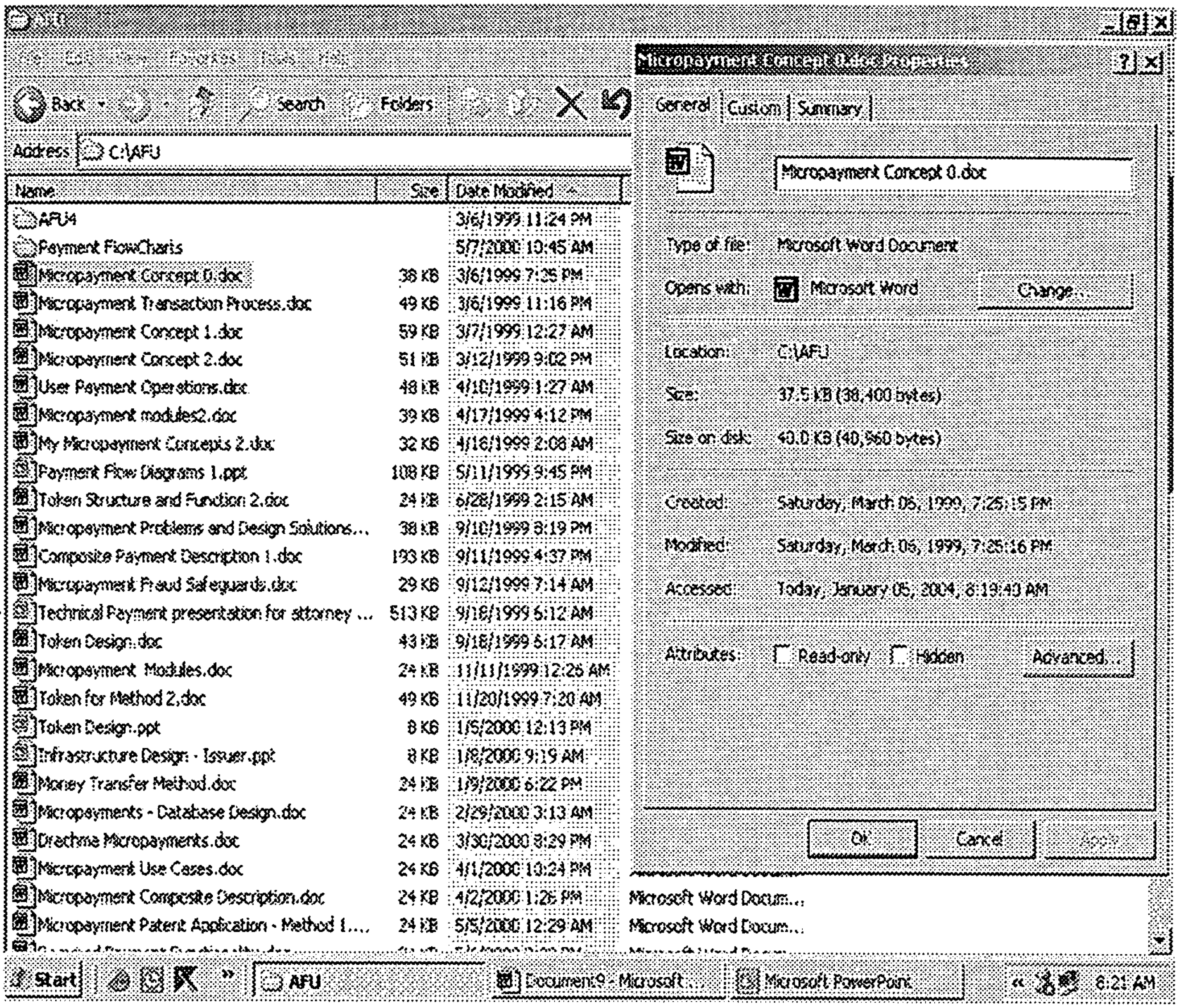
each document was recently prepared to identify the name of the computer file and its creation date. This information was obtained from my computer records, and a copy of a screen capture showing this information also is attached. The attached documentation evidences that the subject matter of the pending claims in my application was conceived of prior to the earliest priority filing date of the Ling Publication.

3. I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statement and the like so made are punishable by fine or imprisonment or both, under section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

____
Chijioke Chukwuemeka

____
(city and state of execution)

Dated: 1/5/________, 2004


AFU
Micropayment Concept 0.doc Properties
Back
Search
Folders
Address C:\AFU
Name
Size
Date Modified
AFU4 3/6/1999 11:24 PM
Payment FlowCharts 5/7/2000 10:45 AM
Micropayment Concept 0.doc 38 KB 3/6/1999 7:25 PM
Micropayment Transaction Process.doc 49 KB 3/6/1999 11:16 PM
Micropayment Concept 1.doc 59 KB 3/7/1999 12:27 AM
Micropayment Concept 2.doc 51 KB 3/12/1999 9:02 PM
User Payment Operations.doc 48 KB 4/10/1999 1:27 AM
Micropayment modules2.doc 39 KB 4/17/1999 4:12 PM
My Micropayment Concepts 2.doc 32 KB 4/18/1999 2:08 AM
Payment Flow Diagrams 1.ppt 108 KB 5/11/1999 9:45 PM
Token Structure and Function 2.doc 24 KB 6/28/1999 2:15 AM
Micropayment Problems and Design Solutions… 38 KB 9/10/1999 8:19 PM
Composite Payment Description 1.doc 193 KB 9/11/1999 4:37 PM
Micropayment Fraud Safeguards.doc 29 KB 9/12/1999 7:14 AM
Technical Payment presentation for attorney … 513 KB 9/18/1999 6:12 AM
Token Design.doc 43 KB 9/18/1999 6:17 AM
Micropayment Modules.doc 24 KB 11/11/1999 12:26 AM
Token for Method 2.doc 49 KB 11/20/1999 7:20 AM
Token Design.ppt 8 KB 1/6/2000 12:13 PM
Infrastructure Design - Issuer.ppt 8 KB 1/8/2000 9:19 AM
Money Transfer Method.doc 24 KB 1/9/2000 6:22 PM
Micropayments - Database Design.doc 24 KB 2/29/2000 3:13 AM
Drachma Micropayments.doc 24 KB 3/30/2000 8:29 PM
Micropayment Use Cases.doc 24 KB 4/1/2000 10:24 PM
Micropayment Composite Description.doc 24 KB 4/2/2000 1:26 PM Microsoft Word Docum…
Micropayment Patent Application - Method 1…. 24 KB 5/5/2000 12:29 AM Microsoft Word Docum…
General
Custom
Summary
Micropayment Concept 0.doc
Type of file: Microsoft Word Document
Opens with: Microsoft Word
Change…
Location: C:\AFU
Size: 37.5 KB (38,400 bytes)
Size on disk: 40.0 KB (40,960 bytes)
Created: Saturday, March 06, 1999, 7:25:15 PM
Modified: Saturday, March 06, 1999, 7:25:16 PM
Accessed: Today, January 05, 2004, 8:19:40 AM
Attributes: Read-only Hidden
Advanced…
OK
Cancel
Apply
Start
AFU
Document9 - Microsoft…
Microsoft PowerPoint
8:21 AM

My Micropayment Concepts

The problems associated with micropayments are soluble in software. i.e., a software solution running on a general purpose computer. Many existing payment systems seem to emphasize hardware, like Mondex which uses a smart-card. These I believe, will fail commercially as the requirement for specialized hardware will make adoption very difficult. I also do not believe that asymmetric encryption is needed as this consumes endless computing cycles and places a ceiling on the number of transactions that can be completed using current hardware.

I have 2 alternative solutions each of which requires a token and involves 3 parties:

a) A token issuer. This is the company – my company – which will issue the tokens.
b) A buyer or consumer who uses the token
c) A merchant who sells items to the buyer.

The Token
The solutions use a digitally encrypted file called a token, which represents cash. The token is downloaded either to the merchant or consumer whenever a transaction is requested. Once the token is received by either the consumer or the merchant subsequent transactions involve only these two parties. The resulting reduction in overhead cost is considerable. The token will have features that make it secure for use on a general purpose computer:

1. It will have encrypted data on it, which the issuer decrypts and reads to authenticate the token each time it is presented;
2. It will expire after a short time (about 2 hours). This expiration life is modifiable;
3. It will reside in Random Access Memory (RAM) and will therefore be difficult to access. Each time the consumer downloads a token from the issuer or transacts with a different merchant, certain fields on the token change. The token is therefore dynamic and impossible to duplicate
4. It will disappear when the consumers computer is turned off. To re-obtain the token the consumer will have to re-connect to the issuer's server and will download a token which will be different from the previous one: It will have a different random; a different expiry time; number amongst other fields from the last one downloaded. The new random means the token will also require a new key with which to modify it.
5. It will hold authentication information as well as its current balance. This allows a display of transaction information in real time.

Each time the consumer downloads a token from the issuer or transacts with a different merchant, certain fields on the token change. The token is therefore dynamic and impossible to duplicate. One of the fields which changes is a random number field. This random number is known only to the issuer and used to generate the key with which to decrement or increment that particular token.

The 3 parties

1. For the first payment method, the consumer downloads a token to his computer after he makes money payment. In other words the consumer "buys" the token. The token therefore represents stored money value. Online payment for the token will be with conventional money instruments such as a debit card, credit card, or electronic check. The consumer can also purchase offline, a prepaid card which resembles a phone card as it represents a token already created by the issuer. The token number or ID of the token when entered at the issuer's server or website activates the token.

2. The consumer receives a either a unique login name and password or a unique token number and PIN for the token purchased. (Going forward I will refer to them interchangeably as token number and PIN or consumer login and password) The consumer enters this token number and PIN when beginning a purchase at a merchant site.

3. In the first method where the token is sent to the consumer, the merchant confirms the genuineness of the token by presenting a copy, received from the consumer, to the issuer along with the merchant's ID (obtained earlier during registration). If the token is genuine, the merchant downloads a key with which to modify the original token still residing on the consumer's machine. With a single key the merchant can perform multiple modifications of a particular token making it unnecessary to keep returning to the issuer. This solves problem of cost overhead associated with small payments. Since the token always has its balance, the invention allows display of a running balance of the token's dollar value as well as the dollar amount of individual transactions in real time. I know of no other transaction instrument – save cash – which gives your balance in real time.

4. The second micropayment method is like the first in that a token is created when the buyer pays for it. So the token also represents stored money value. The difference is that the token is not sent to the buyer but to the merchant where the buyer wishes to make a purchase. Here, when the buyer wants to make a purchase, he again enters a token number and PIN. The merchant submits this token number and PIN to the issuer along with his own unique merchant ID (obtained earlier during registration). The issuer downloads to the merchant, the buyer's token along with a key with which to modify it. Here again, after the download, the merchant can modify the token locally for multiple transactions without having to return to the issuer each time for authorization. Both cases require that the merchant "register" with the issuer and obtain a unique ID as well as software that will handle decrementing the token, contacting the issuer, downloading the buyers token, storing transaction records, and periodically uploading the token and the token transaction records to the issuer.

Differences between the two methods

The first method is called buyer-hold because the buyer holds the token and the second method is merchant-hold as the merchant holds the token.

Buyer-hold

The buyer-hold will have greater application for offline purchases using devices not connected to the internet such as PDA's, smart cards. Although typically the token has a short life (about 2 hours) this will be increased to 24-48 hours for offline devices. In addition the token can after which it expires and a new token would have to be obtained by contacting the token issuer. Also the token With hand held devices that are not internet connected, the life of the token can be lengthened and it is made "persistent" i.e. After it is downloaded into the consumer's device it does not disappear when the device is switched off. The merchant writes to – modifies – the token with each transaction. Once a token is made "persistent" it cannot receive credits. This is a safety measure.

Merchant-hold

Merchant-hold will have greater application for online transactions using devices permanently connected to the internet.

Either method even though designed especially for micropayments can obviously be used for ANY payment amounts. In pursuing intellectual property protection, both methods will be named electronic payment methods as they are not limited to micropayments.

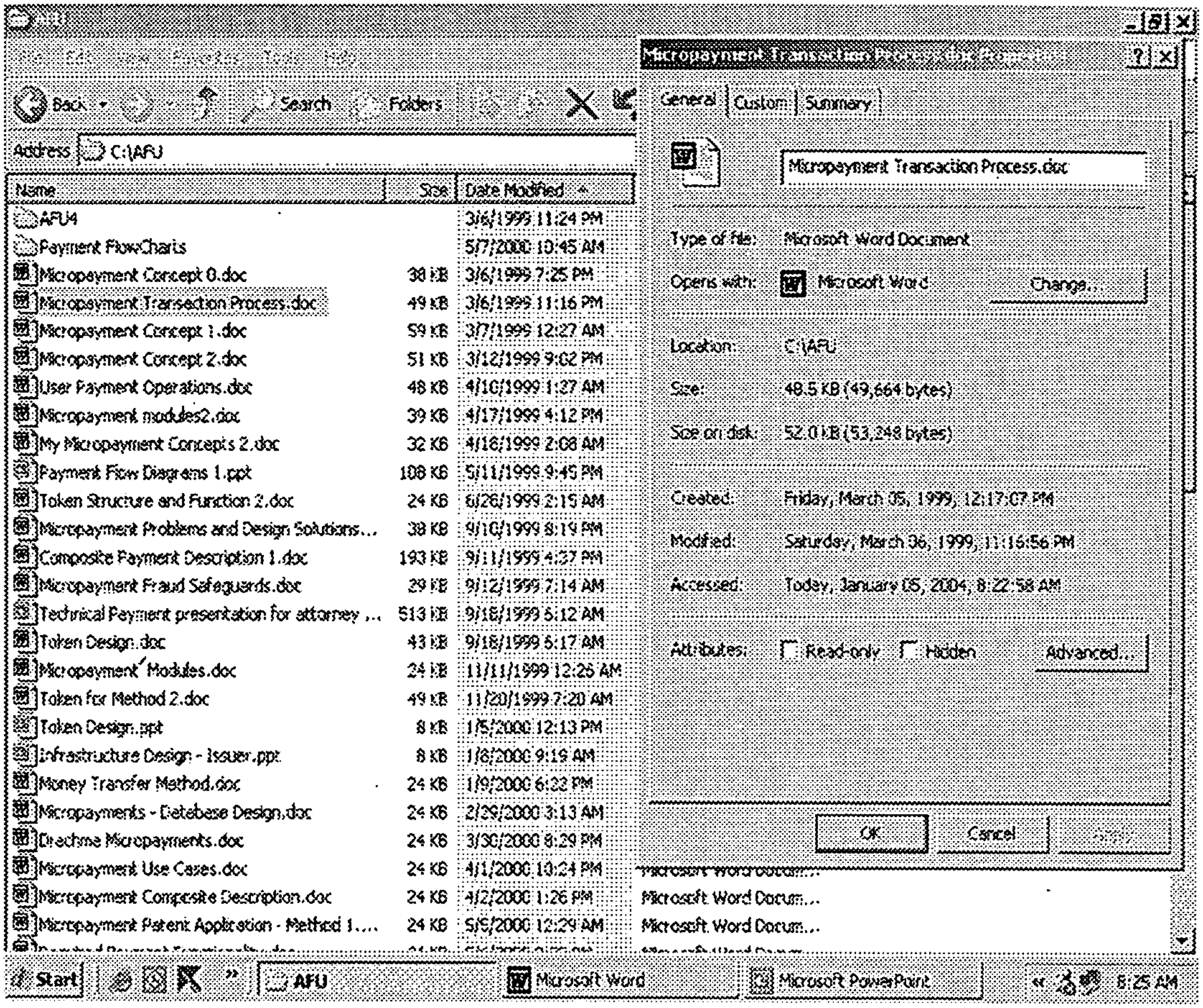
What my methods solve

Each method solves the key problems of micropayments

1. Minimal overhead costs for multiple transactions. Once the token or key is downloaded all subsequent transactions are "local" and involve no further contact with the issuer.
2. Permit real-time display of running balance
3. Maintain security even though transaction amounts are small
4. Permit credits to be applied to the token
5. Allow transfers of funds between tokens as this can be viewed as a debit of one token and a credit to, or creation of another token.
6. Can be generalized for use for any payment amounts since this is valid for any amounts to any number of decimal places. e.g., 1.23367 cents or $1,000,000
7. Require only a token for transactions to be completed. The merchant cannot know the identity of the buyer
8. Allows the buyer to buy a Token offline using a prepaid card. This card represents a pre-created entry in the issuers token records.

I expect to file for patents on both processes but will probably file one first and then the other later except perhaps if I can get reduced costs on 2 filings. It may also be possible to fold them into one patent although I would rather have them considered 2 separate ideas which should produce 2 separate patents.

I will give details of the buyer-hold method below. I have been told to be consistent in my naming so I instead of Issuer, going forward, I will use Clearing server.


Micropayment Transaction Process.doc Properties
General
Custom
Summary
Micropayment Transaction Process.doc
Type of file: Microsoft Word Document
Opens with: Microsoft Word
Change...
Location: C:\AFU
Size: 48.5 KB (49,664 bytes)
Size on disk: 52.0 KB (53,248 bytes)
Created: Friday, March 05, 1999, 12:17:07 PM
Modified: Saturday, March 06, 1999, 11:16:56 PM
Accessed: Today, January 05, 2004, 8:22:58 AM
Attributes: Read-only Hidden Advanced...
OK
Cancel
Back
Search
Folders
Address C:\AFU
Name
Size
Date Modified
AFU4 3/6/1999 11:24 PM
Payment FlowCharts 5/7/2000 10:45 AM
Micropayment Concept 0.doc 38 KB 3/6/1999 7:25 PM
Micropayment Transaction Process.doc 49 KB 3/6/1999 11:16 PM
Micropayment Concept 1.doc 59 KB 3/7/1999 12:27 AM
Micropayment Concept 2.doc 51 KB 3/12/1999 9:02 PM
User Payment Operations.doc 48 KB 4/10/1999 1:27 AM
Micropayment modules2.doc 39 KB 4/17/1999 4:12 PM
My Micropayment Concepts 2.doc 32 KB 4/18/1999 2:08 AM
Payment Flow Diagrams 1.ppt 108 KB 5/11/1999 9:45 PM
Token Structure and Function 2.doc 24 KB 6/26/1999 2:15 AM
Micropayment Problems and Design Solutions... 38 KB 9/10/1999 8:19 PM
Composite Payment Description 1.doc 193 KB 9/11/1999 4:37 PM
Micropayment Fraud Safeguards.doc 29 KB 9/12/1999 7:14 AM
Technical Payment presentation for attorney ... 513 KB 9/18/1999 6:12 AM
Token Design.doc 43 KB 9/18/1999 6:17 AM
Micropayment Modules.doc 24 KB 11/11/1999 12:26 AM
Token for Method 2.doc 49 KB 11/20/1999 7:20 AM
Token Design.ppt 8 KB 1/5/2000 12:13 PM
Infrastructure Design - Issuer.ppt 8 KB 1/8/2000 9:19 AM
Money Transfer Method.doc 24 KB 1/9/2000 6:23 PM
Micropayments - Database Design.doc 24 KB 2/29/2000 3:13 AM
Drachma Micropayments.doc 24 KB 3/30/2000 8:29 PM
Micropayment Use Cases.doc 24 KB 4/1/2000 10:24 PM
Micropayment Composite Description.doc 24 KB 4/2/2000 1:26 PM
Micropayment Patent Application - Method 1.... 24 KB 5/6/2000 12:29 AM
Microsoft Word Docum...
Microsoft Word Docum...
Start
AFU
Microsoft Word
Microsoft PowerPoint
8:25 AM

The Micropayment Problem. My Insights and Solutions

All payment technologies are of 2 types: notational systems and value systems.

A. Notational systems are best represented by payment by check and involve 3 parties: A consumer, a merchant and a bank. A consumer pays a merchant by check, the merchant deposits the check at his bank. His bank presents the check to the consumer's bank. The consumer's bank debits the consumer's account, credits the merchant's account--that is the check is cleared--and informs the merchant who can then release the goods. The notational system has a lot of overhead and is therefore unsuitable for micropayments. The time overhead is so great that even for brick and mortar transactions the merchant typically collects customer identifying information and releases the sold goods before the consumer's check clears.

B. Value systems use a digitally encrypted message, called a token, which holds monetary value. The token, is presented by the consumer to the merchant to make a purchase. The merchant ascertains that the token is authentic, usually by decrypting the encrypted signature of the issuer, and can thereafter debit the token directly for a sale without having to contact a third party, (the bank issuer). The overhead costs are lower since no third party has to be consulted and this makes value systems the method of choice for micropayments.

<u>Problems with Value systems</u>. Value systems require that a token representing an encrypted message hold money value which can be transferred to a merchant without the need to contact the issuer of the token. This presents several problems:

1) <u>Double spending</u>. Since the token is digital, the consumer can, in theory, make numberless copies and present each copy to a different merchant. Each copy is a perfect replica of the original and will pass any authentication tests. In other words, the consumer can spend the same token two or more times. This is the "double spending problem". Several solutions are plausible.
    a. The merchant could contact the token issuer for every transaction to confirm that the token received has not been "spent" before. This re-creates the overhead problem of the Notational system, since the merchant must contact the issuer for every transaction. In addition, the issuer must have continuously updated records of all tokens outstanding (issued to consumer) and all tokens redeemed (spent by consumer and returned to the Issuer by the merchant) . The storage requirements are extremely large as is the computational load on the issuer. A token for example which was last spent 9 months ago must be retrieved in response to a merchant inquiry. If the token was last spent 5 years ago, the time taken for retrieval becomes unacceptably large. If the issuer is processing several requests, the issuer will be unable to handle very many transactions. Preventing double spending through confirmation with the token issuer also assumes that the merchant immediately after he receives an authentic token from a sale, instantly deposits it with the issuer. If there is a time lag between receiving a genuine payment and depositing it with the issuer, then double spending can still take place. The fraudulent consumer simply makes a purchase at a merchant and before that merchant deposits the token, makes a second purchase at another merchant with a copy of the token. Since the first merchant has not deposited the original token with the issuer, an inquiry to the issuer by the second merchant will confirm that the token is real, was issued and has not been redeemed.

b. A more common solution is to the double spending problem is to introduce a hardware device that would hold the token and make it inaccessible to the consumer and therefore impossible to copy. Transactions would require that the merchant access the token within this hardware device, authenticate and then debit the token to conclude a sale. This requires that both the consumer and the merchant use a special device into which the token would be downloaded. The token will therefore be inaccessible to the consumer and therefore cannot be copied. This solves the double spending problem but means that micropayment transactions, would require every participant, consumer or merchant to have a smart card, which, in turn, makes adoption insuperably difficult. All hardware solutions imply a very high adoption hurdle and an exponentially high likelihood of commercial failure.

2) <u>The Change problem</u>. Assume that double spending was somehow solved in software, other problems remain. Imagine a scenario where a consumer presents a $10 token and wishes to purchase a 5¢ item. The merchant needs to "create" a new token with value $9.95 and ensure that it has all the security and acceptability features of the original $10 token. So to make change each merchant issues new cash. All participants in such a system must agree a priori to issue currency and accept each others currencies. Some solutions to the change problem require the consumer to carry multiple tokens (of one issuer) in several denominations. It is easy to derive the minimum number of tokens required in order to make a payment of any amount below $1 for example. The problems is that after any purchase, the remaining number of tokens can no longer be used to pay for any amount and the consumer must replace the depleted tokens after each transaction. A consumer with 5 pennies, 1 nickel, 4 dimes and 2 quarters can make payment for any article that costs a dollar or less without requiring change. The consumer may for example be able to buy a 49¢ item once but thereafter will be unable to purchase a 38¢ item without replacing his depleted dimes. Token replacement, after each transaction increases overhead and means again that small transactions become impractical.

3) <u>The Money Value problem</u> For the token to be used in purchases, it must have money value attached to it. The options are for the consumer to "pay" with conventionally accepted currency in order to create a token that is backed by money or for a consumer or issuer to create "currency" not backed by money but which is redeemable for money after it is spent. In the latter case, a mechanism must exist to charge the consumer and credit the merchant post-transaction. It also requires all transacting parties to believe that the "currency" they create can be redeemed for cash. This creates problems of convertibility, and requires the issuer to be a "Central Bank". This makes adoption very difficult.

4) <u>The Security problem</u> Providing security for small transactions has proven intractable for many operators. Most payment systems use Private/Public Key (PPK) encryption. This is a technique where the encryption key and the decryption keys are different. One party, usually the issuer, distributes his public key widely and encrypts tokens with his private key (which no other party knows). The public key can then be used to decrypt any token written with the issuers private key. The problem is that PPK is computational intensive and requires each merchant to possess a super computer in order to process a moderate number of transactions. People have tried to reduce the computation load by not encrypting the entire token but encrypting instead a numeric value obtained by running the token through a specific algorithm. The recipient of the token, decrypts the encrypted hash value and then runs the token through the same algorithm. It the value obtained matches the value decrypted, the token is assume to be authentic. Using hashes as opposed to encrypting the whole token only moderately reduces the computational load. Dropping the encryption requirement

compromises security. In fact some, micropayment methods require the consumers and merchants to be willing to lose money on their payment transactions.

I believe that the problems encountered in solving the Micropayment problem are very similar to the problems encountered when man first tried to fly. The idea was to imitate birds by obtaining wings that flap. Flight was only perfected when that idea was abandoned and instead the Wright Brothers pursued the idea of staying airborne, which they realized could be done without flapping wings. In the same way, most digital payment systems are trying to duplicate the transactional operations of currency: You give a merchant cash – the palpable hard coins or notes that you feel in your wallet -- and the merchant gives you the item you purchased with any change due. This effort to replicate the operations performed in a cash transaction are both difficult and unnecessary in a digital medium. Digital money can and should perform the exchange and storage functions of money without replicating the look, feel or operational steps performed with physical cash.

My solution reduces overhead, enables scalability and ensures security without raising transaction cost. I describe the conceptual basis of my digital payment method below.

1) CORE CONCEPT
   a. Have a single token that is really a pointer to information that resides elsewhere such as on a central server. The token simply holds at a minimum a unique number and PIN. The token number and PIN is the minimum required to retrieve the actual records for that token which reside on a remote server. These records include the balance on the token, what transactions were performed on the token, by whom the transactions were performed and whether the token is currently "checked out" i.e. residing with a merchant , or if it is held by the central server. For each transaction, a part of the token record moves to the transaction location. I have 2 solutions: One where the token record moves to the merchant and another completely different solution where it moves to the consumer. I will file patents for both solutions.

   b. In either, case debits and credits occur at the transaction location between the merchant and the consumer. When the transactions are completed the record for that token is returned to the central server.

2) TOKEN MOVES TO MERCHANT
   a. In the case where, the token is sent to the merchant, the consumer enters the token number and PIN in order to complete a purchase. The merchant forwards this number and PIN to the central which first authenticates the merchant and then sends him the token record with authority to decrement or increment the balance on the token and record the transactions associated with each increment and decrement. Once the token arrives at the merchant, (i.e. merchant merchant's computer), he can debit or credit it multiple times without the need to contact the central server. The token once it is downloaded is "local" and so there is very little overhead cost in processing subsequent transactions, small or large after the initial download to the merchant. The merchant has to have registered beforehand with the central server and obtained software which does processing on the merchant's server.

   b. The central server simply authenticates the requesting merchant, identifies the record associated with the token number and PIN, updates the records associated with that

token if needed (the token record may currently be "checked out" i.e., residing with another merchant and will have to be retrieved), and then downloads the token record with a key to the requesting merchant. The central server is not burdened with processing actual transactions and can therefore handle a very, very large number of token requests. Since actual processing occurs at the merchant's computer this represents a distributed solution and means that transaction processing can scale i.e., that is the number of transactions processed can be increased almost without limit.

3) TOKEN MOVES TO CONSUMER.

   a. In the case where processing is done on the consumers device, the same advantages described above also accrue. The processing is distributed, the transactions scale.

   b. The consumer has to first purchase and download a token along with software that manages the token on the consumers computer including: incrementing and decrementing the token when a key is received from a merchant, keeping the token in RAM, destroying the token when it expires etc. The consumer downloads this software with the token from the central server when first registering.

   c. The consumer presents this token to the merchant in order to perform a transaction. The consumer may be required to enter a PIN number for additional security.

   d. The merchant forwards the consumer's token to the central server which authenticates and gives the merchant a key that authorizes the merchant to modify (decrement or increment) the value of the token on the consumer's computer. The modification is performed when the merchant presents a request for a decrement or increment along with a key to the software on the consumer's computer. It is actually the consumer's software which does the decrement/increment of the token, although the merchant's software also records the amount of the decrement and the specific token involved. It is the merchant's token data which is used to update the token.

   e. The consumer can then perform several transactions with that merchant who can decrement the token immediately without having to contact the central server. The merchant keeps a record of the transactions performed on the token and uploads these transactions to the central server periodically or in response to a poll from the central server. Once a token record is uploaded no further transactions can be performed on the token, unless the merchant makes another request for the token and update key along with the upload.

   f. Consumer transactions with a merchant a described as "sessions". A session is an uninterrupted period of transactions with ONE merchant, without any intervening transactions with another merchant and before expiration of the token. During a session, the merchant has the key and the merchant can continue modifying the token without contacting the clearing server.

   g. If the consumer should move from the first merchant to a second merchant to transact, he again submits his token, to the second merchant, who again forwards this token to the clearing server. This time the clearing server notices that a token key was given to the first merchant who has not yet sent an upload. It polls the first merchant obtains token updates, recalculates the balance on the token, creates a new token and then downloads this new token with 2 keys (they could really be one key)

an update key and an overwrite key, to the second merchant. This is a second unique "session".

h. This second merchant then overwrites the consumer's old token with the new one and can then begin decrementing or incrementing the token with the second key. The primary advantage of delivering the token to the consumer is that it can be used for transactions offline as the consumer does not require a connection to the Internet or a network. All that is required is connection to the merchant's computer which can be achieved either by a direct physical connection, infra-red, blue tooth, short range wireless, etc. Since the token holds a running balance of transactions performed, the consumer also obtains a real time balance of transactions. The problems are that since the token itself is the primary authorization required to receive a key, it must have security features which make it difficult to duplicate.

i. The security features are:

    i. The token resides in RAM on the consumer's computer and disappears when the consumer's computer is switched off. The consumer has to contact the clearing server to re-obtain the token in this case. This feature can be turned off for portable handheld devices, so that the consumer can turn off the device turn it back on and still have the token. Since the token is modified by the merchant as the consumer shops, the consumer can use the device to shop offline, while still obtaining a running transaction balance. The merchant's however have to be connected to the clearing server. The capacity to obtain a real-time balance while transacting is not as far as I know implemented in any payment system apart from cash.

    ii. The token has a short life and expires after a few hours. This can be modified but is a safety precaution. If the token expires while on the consumer's computer it is removed from RAM by the consumer's software. If it expires while the consumer is transacting with a merchant, the merchant uploads all the records and requests a new token and a new key.

    iii. Every time the token is re-obtained it has a different number known only by the clearing server as well as modified fields, including date and expiration. Each token is therefore different. The token fields are stored on the central server. A token presented by a merchant has to match the stored fields which makes duplication impossible.

    iv. Although the token itself is likened to money, it is really an authentication key which the merchant forwards to the central server.

4) SCALING THE CENTRAL SERVER

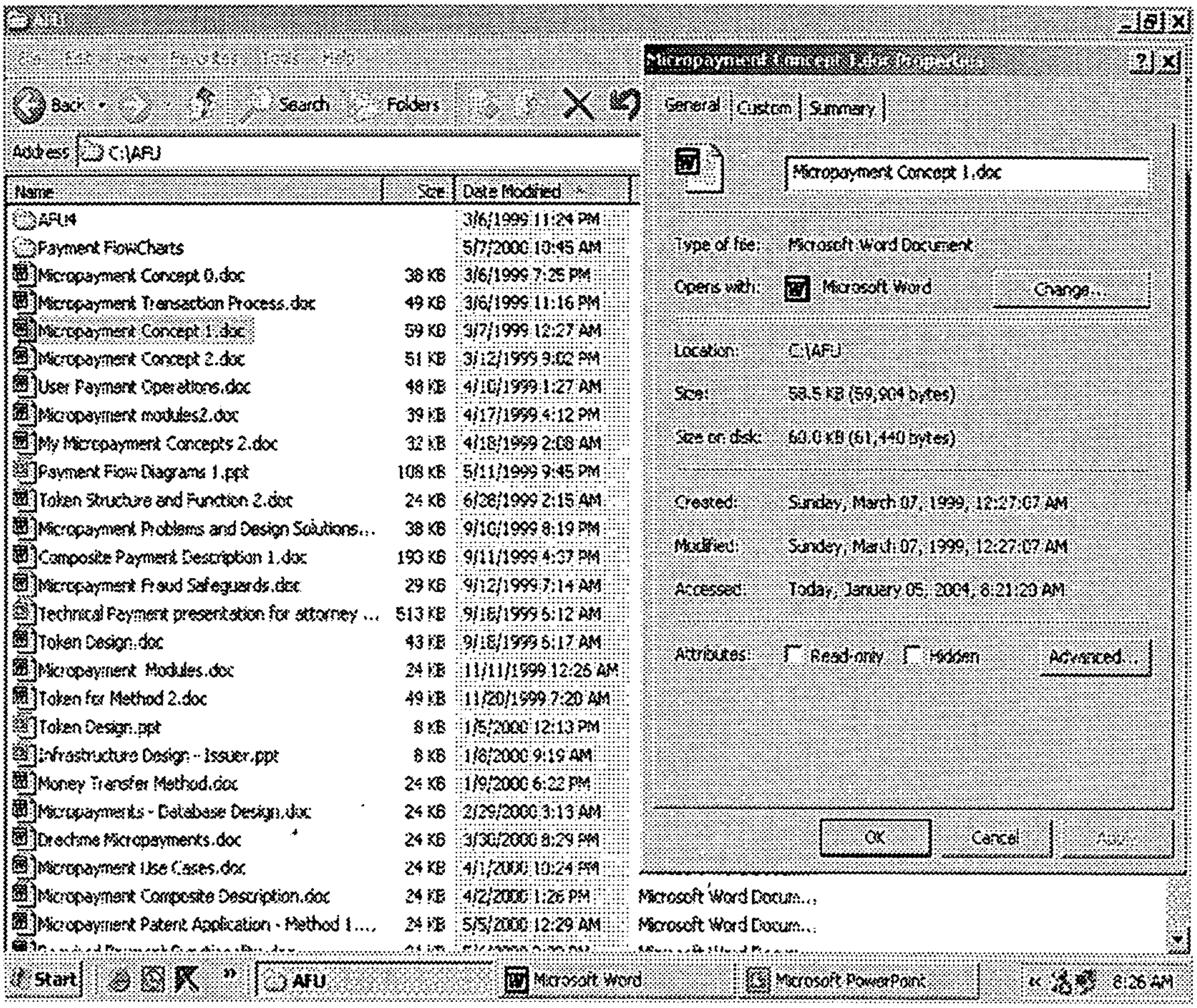
a. There will be several central servers that hold token records and there are very many schemes to divvy up which servers hold which token records. These include breaking the tokens into ranges with different servers holding different records. The idea is for each central server to generate a token number in a way which uniquely identifies that token as belonging to that central server. In other words, there will be a number of token central servers with responsibility for token records in certain

number ranges. This is similar to but not identical with, and certainly not the same as, the domain name system (DNS).

b. The token Number could for example be a 17 digit number (It could also be alpha numeric to increase the range). This gives a set of 10^{17} possible numbers. Each central server will have a range of numbers within this set for which it is responsible for example, the first central server could own the range of 17 digit numbers beginning with 1and 2. Another server would own the range of 17 digit numbers beginning with 3 and 4 and so on. Each central server would in turn have 0 or more subordinate servers who would own records within the number subset handled by their parent central server. These subordinate servers will therefore only create tokens with numbers within the number subset allocated to the central server. The central server will determine the portions of its number subset to be handled by the various subordinate servers.

c. A concrete example. A consumer submits to a merchant the token number 1234 5678 9012 34567 and PIN. The central server that receives this request serve only token numbers beginning with 3 and 4, so it forwards the token request to the central server responsible for numbers 1 and 2. This server receives the request and then checks its internal tables, to identifies the subordinate server that handles token number 1234 and forwards this request to the server. The subordinate server may in turn process the request or forward it to its own subordinate servers to whom it delegates portions of its own number subset, perhaps a server handling tokens with numbers in the range 1234 5xxx to 1234 6xxx and so on. The server which actually holds the token record then forwards it either directly to the merchant or to the initial requesting central server or to the server where the merchant's registration record resides, for onward delivery to the merchant. The merchant performs transactions on the token record and then returns it to its central server either as part of its periodic upload or in response to a poll. I'll expand on this distributed central server processing in a separate document. The point is that it means that my payment method can scale to handle billions of transactions, and will not be limited by geography.

5) The download of the token to the merchant requires that the merchant be registered beforehand with the central server and that the merchant receive software from the central server. This merchant will present a key which was received at registration along with its request for a token record. Initially "handshake" between the clearing server and the merchant can be performed with this key and will make merchant authentication much faster as the central server does not have to first negotiate and agree on a unique key as required in Secure Socket Layer (SSL) transactions.


Back
Search
Folders
Address C:\AFU
Name
Size
Date Modified
AFU4 3/6/1999 11:24 PM
Payment FlowCharts 5/7/2000 10:45 AM
Micropayment Concept 0.doc 38 KB 3/6/1999 7:25 PM
Micropayment Transaction Process.doc 49 KB 3/6/1999 11:16 PM
Micropayment Concept 1.doc 59 KB 3/7/1999 12:27 AM
Micropayment Concept 2.doc 51 KB 3/12/1999 9:02 PM
User Payment Operations.doc 48 KB 4/10/1999 1:27 AM
Micropayment modules2.doc 39 KB 4/17/1999 4:12 PM
My Micropayment Concepts 2.doc 32 KB 4/18/1999 2:08 AM
Payment Flow Diagrams 1.ppt 108 KB 5/11/1999 9:45 PM
Token Structure and Function 2.doc 24 KB 6/28/1999 2:15 AM
Micropayment Problems and Design Solutions... 38 KB 9/10/1999 8:19 PM
Composite Payment Description 1.doc 190 KB 9/11/1999 4:37 PM
Micropayment Fraud Safeguards.doc 29 KB 9/12/1999 7:14 AM
Technical Payment presentation for attorney ... 513 KB 9/16/1999 6:12 AM
Token Design.doc 43 KB 9/18/1999 6:17 AM
Micropayment Modules.doc 24 KB 11/11/1999 12:26 AM
Token for Method 2.doc 49 KB 11/20/1999 7:20 AM
Token Design.ppt 8 KB 1/5/2000 12:13 PM
Infrastructure Design - Issuer.ppt 8 KB 1/8/2000 9:19 AM
Money Transfer Method.doc 24 KB 1/9/2000 6:22 PM
Micropayments - Database Design.doc 24 KB 2/29/2000 3:13 AM
Drechme Micropayments.doc 24 KB 3/30/2000 8:29 PM
Micropayment Use Cases.doc 24 KB 4/1/2000 10:24 PM
Micropayment Composite Description.doc 24 KB 4/2/2000 1:26 PM
Micropayment Patent Application - Method 1.... 24 KB 5/5/2000 12:29 AM
Microsoft Word Docum...
Microsoft Word Docum...
Micropayment Concept 1.doc Properties
General
Custom
Summary
Micropayment Concept 1.doc
Type of file: Microsoft Word Document
Opens with: Microsoft Word
Change...
Location: C:\AFU
Size: 58.5 KB (59,904 bytes)
Size on disk: 60.0 KB (61,440 bytes)
Created: Sunday, March 07, 1999, 12:27:07 AM
Modified: Sunday, March 07, 1999, 12:27:07 AM
Accessed: Today, January 05, 2004, 8:21:20 AM
Attributes: Read-only Hidden
Advanced...
OK
Cancel
Start
AFU
Microsoft Word
Microsoft PowerPoint
8:26 AM

My Micropayment Invention

The problems associated with micropayments are soluble in software. i.e., a software solution running on a general purpose computer. Many existing payment systems seem to emphasize hardware, like Mondex which uses a smart-card. These I believe, will fail commercially as the requirement for specialized hardware will make adoption very difficult. I also do not believe that asymmetric encryption is needed as this consumes endless computing cycles and places a ceiling on the number of transactions that can be completed using current hardware or any hardware.

I have 2 alternative inventions. Each invention involves 4 parties:

a) A creditor or Banking institution through which payment is made for the Token thereby ensuring it is backed by money
b) A clearing server. This server will authenticate tokens and issue keys with which the tokens can be modified: decremented, incremented or overwritten. Tokens are digital representations of cash.
c) A buyer or consumer who uses the Token
d) A merchant or vendor who sells items to the buyer.

The process in summary will be as follows:

1. In the first invention, a buyer downloads a token to his computer after he makes money payment. In other words the buyer "buys" the token. The token therefore represents stored money value. The Token "purchase" is performed through a clearing server which contacts the Bank or creditor, obtains the purchase amount and then creates the Token. The Token is not strictly a purchase since it can be redeemed for its cash value. The buyer can then visit a merchant and make purchases paid for with this "local" token. The merchant modifies the token after each sale using a key downloaded from the clearing server. The key is specific to each token. With a single key the merchant can perform multiple modifications (increment or decrement) of a particular token making it unnecessary to keep returning to the clearing server. This solves problem of cost overhead associated with micropayments (see description of micropayment systems below). The token also has characteristics which make it secure for use on a general purpose computer: It disappears when the buyer's computer is turned off; expires after a fixed period of time (which is changeable); can only be modified on presenting a key; contains authentication information; and can display, the current balance as well as the amount of each individual transaction in real time. I know of no other transaction instrument – save cash – which gives your balance in real time.

2. My second micropayment method is similar to the first in that a Token is created when the buyer pays for it. So the Token also represents stored money value. The difference is that the Token is NOT sent to the buyer but sent to the merchant whenever the buyer wishes to make a purchase. Here, when the buyer initiates a purchase, the merchant contacts the clearing server and downloads the buyer's Token

along with a key with which to modify it. Here again the merchant can modify the token locally for multiple transactions without having to return to the clearing server each time for authorization.

Either invention requires that the merchant "register" with the clearing server and obtain software that will handle contacting the clearing server, downloading the buyer's Token, modifying the Token, storing Token transaction records, and periodically uploading the Token with the Token transaction records to the clearing server. The merchant will also have to provide his URL, email and bank account information so that they can be contacted and so that their funds at the clearing server can be transferred to their bank account. The first invention also requires the consumer to download software to his computer. The software keeps the token in memory, modifies the token when a merchant presents a valid key with an increment or decrement request, displays a running balance of the token, and keeps a record of transactions by merchant.

Each invention requires that each Token be unique and that each buyer transact with only ONE merchant at a time. This prevents fraud. An example shows why.

a) A buyer may try to make simultaneous purchases from 2 different merchants, at the same time, for example buy a $3 item from two different merchants, merchant1 and merchant2, at the same time with a single $4 dollar token.
b) The request from merchant1 arrives first at the clearing server and is processed first. Note that time is in milliseconds and even if the requests arrive at "exactly the same time" the clearing server will arbitrarily chose one request to process first.
c) Merchant1 receives a key from the clearing server, and decrements the buyers token by $3.
d) Then the request for the merchant2 is received.
e) The records from the merchant1 are uploaded to the clearing server, the token balance is updated to reflect the new balance of $1
f) A new $1 token with an overwrite key is downloaded to merchant 2.
g) Merchant 2 alerts the buyer that he has $1 and cannot make a $3 dollar purchase.

This makes it impossible for a buyer to spend more money than is held by the Token. Although the clearing server performs transactions in a linear sequence, high transaction speeds will give the illusion of simultaneity as happens in CPU context switching.

<u>**Applications for Each Invention**</u>
I have called the first invention **buyer-hold** payment method because the buyer holds the token and the second invention is called **merchant-hold payment** method as the merchant holds the token. These 2 inventions though similar are not variants but use significantly different approaches to solving the micropayment problem. With buyer-hold, the software on the buyer's computing device performs the actual debit (or credit of the token) on behalf of the merchant once the merchant presents the correct key. In merchant-hold, the merchant receives the token and debits (or credits) the token directly on receiving confirmation from the buyer. Each method therefore has different security

implications and different requirements for implementation even though they may appear cognate.

Buyer-hold
The buyer-hold will have greater application for offline purchases using devices NOT connected to the internet such as PDA's, smart cards. In this case the life of the token is lengthened and it is made "persistent" i.e. After it is downloaded into the consumer's device it does not disappear when the device is switched off. The merchant authorizes modification of the token for each transaction by presenting a key downloaded from the clearing server. Once a Token is made "persistent" it cannot receive credits. This is a safety measure. The persistence feature is requested by the buyer and turned on at the clearing server. This means that a buyer can make the token persistent i.e., give it a life of several hours, download the token into a hand held device – for example a palm pilot – and then shop offline. Each merchant where the buyer wishes to make a purchase downloads a key from the clearing server and can thereafter modify the Token for each purchase. Of course persistence is not required if the buyer is willing to leave her computing device on for long periods.

Merchant-hold
Merchant-hold will have greater application for online transactions using devices permanently connected to the internet. A buyer using a cell phone for example need not download into the cell phone. In this case the buyer again enters a unique token identification number and PIN (login and password) both of which the merchant submits to the clearing server. The token is then downloaded to the merchant. The merchant must have previously "registered" with the clearing server to receive a unique identification number and also obtain software that manages the Token.

Either method even though designed especially for micropayments can be used for ANY payment amounts. In pursuing intellectual property protection, both methods will be named electronic payment methods as they are not limited to micropayments.

What my inventions solve
Each method solves the key problems of micropayments

1. Minimal overhead costs for multiple transactions. Once the token or key is downloaded all subsequent transactions are "local" and involve no further contact with the clearing server.
2. Permit real-time display of token balance after each transaction. The Token is modified after each transaction and can therefore display real time transaction balances.
3. Each transaction is secure regardless of the size of the transaction. Security can be increased by using stronger authentication with little additional overhead.
4. Permits credits to be applied to the token. Here the merchant needs to request a key with which credits can be applied and needs to have a deposit with the clearing server which ensures that the credits applied are backed by real money.
5. Allow transfers of funds between tokens. This can be viewed as a debit of one token and a credit to, or creation of another token.

6. Can be generalized for use for any payment amounts since increments or decrements of a token can be performed for any amounts to any number of decimal places. e.g., 1.23367 cents or $1,000,000.00. Although monetary transactions typically require only 2 decimal places. The token can be used to communicate much larger numbers securely. Using a number to word dictionary would permit the token to be used as a communication device.
7. Require only a token login name and password for transactions to be completed. The merchant cannot know the identity of the buyer.
8. Allows the buyer to buy a Token offline using a prepaid card. This card represents a pre-created record in the clearing servers token database.
9. Allow transactions to scale. Since computations associated with each transaction are performed either at the buyer's computer (method 1) or at the merchant's computer (method 2), this is a distributed model that allows near-limitless numbers of merchants and buyers to transact. The limit is imposed by the need to obtain a key from the clearing server. Since all contiguous transactions at one merchant require only one request for a key from the clearing server, the clearing server does not pose a bottleneck. 10 Contiguous transactions mean that the buyer performs 10 transactions with vendor1 without any intervening transaction with any other and would make only one call to the clearing server.

I expect to file for patents on both processes but will probably file method 1 first and then the method 2 except perhaps if I can get reduced costs on 2 simultaneous filings. It may also be possible to fold them into one patent although I would rather have them considered 2 separate inventions which should produce 2 separate patents.

The are two primary systems underlying payments in general and micropayments in particular. These are notational and value systems. I summarize below how they work and show how my invention differs from these systems.

Notational payment systems resemble purchases by check: A buyer who has a checking account, makes a purchase and pays by check. The vendor confirms that the check is backed by an account, releases goods to the buyer and then deposits the check in his (vendor's) bank account. The vendor's bank forwards the check to the buyer's bank, who then debits the buyer's account, credits the vendor's account at the vendor's bank and returns the paid (cancelled) check to the buyer. Notational systems create large amounts of transaction overhead, are difficult to conclude in real-time and are very expensive to roll back. They are unsuitable for small transactions.

Token-based payment systems are designed primarily to reduce the overhead associated with notational systems. A consumer pays for and downloads into his computing device an encrypted digital file called a Token. The Token is prepaid cash and can be debited directly by a vendor, thereby eliminating the overhead associated with notational transactions. However, the consumer can make multiple perfect copies of a digital Token and spend at least double the money value of the Token. One way to circumvent this is to have each merchant present each token to the issuing authority to confirm that it has not been spent before. This re-introduces the cost overhead of the notational system and

makes the system unscalable as there can only be one issuing authority: If there are multiple issuing authorities responsible for different regions then "double spending" of token copies can still occur in jurisdictions controlled by different issuing authorities.

The "double spending" problem was solved in the past by making the token inaccessible to the consumer primarily by downloading it into a specialized piece of hardware: a smartcard. The vendor also needed to have specialized hardware that would interact with the software. The requirement that the consumer and vendor purchase additional hardware in order to perform transactions made adoption very difficult. A complete software solution, with low overhead costs which also solves the double spending problem was needed but thought impossible. My invention solves both the double spending and the overhead cost problem with software.

My payment invention innovation is a hybrid between Token-based and Notational Payment systems. It is primarily designed for the Internet and involves 4 parties: A bank or creditor, a consumer, a vendor and a clearing server. It works as follow: The consumer purchases a Token through the clearing server, selects a login and password and downloads the Token to his computer along with a small piece of client software. The Token has a fixed life (for example 3 hours), resides in memory on the consumer's computer and is therefore difficult to access or modify. Its cash value which is displayed in a small window changes as purchases are made or credits received. If the Token should expire or if the consumer should switch off his computer, he need only return to the clearing server and enter his login and password to re-acquire his Token. The login name and password are associated with one and only one Token. In other words the login name and password represent unique identifiers for each token. The Token itself has no information on it that reveals the identity of the consumer.

The consumer now goes to a participating vendor, vendor1, clicks on an item to purchase, and (if this is the very first purchase) is prompted to enter his login name and password. The software on vendor1's computer retrieves and sends a copy of the consumer's Token to the clearing server along with a request for a key. The key is unique for each Token and is needed to decrease (or increase) the value of the Token. The clearing server authenticates the Token and then sends vendor1 a key which is used to decrease the consumer's Token for the purchase. The product purchased is then downloaded or sent to the consumer who sees the dollar balance in his Token window decrease by the amount of the purchase. Vendor1 does not need to contact the clearing server for subsequent transactions with this consumer. He simply uses the same key to decrease the consumer's Token for each purchase as long as the Token does not expire and the consumer makes no intervening purchase with another merchant.

If the consumer wishes to make a purchase from a second vendor, vendor2, he connects to vendor2's site and selects an item to purchase. Again he enters his login name and password following which vendor2 forwards along with a key request to the clearing server. Now the clearing server sees that a request was made earlier for a key for this Token by a different merchant, vendor1, so it does the following:

a) polls vendor1,
b) invalidates the key of vendor1 for this Token,
c) uploads from vendor1 records for transactions involving this Token,
d) updates its records to show the new Token balance and then
e) downloads a new updated Token with a unique key to vendor2.

Vendor2 now overwrites the consumer's old Token with the new Token, decreases the new Token with the key, and is now ready to process multiple transactions involving this Token without having to contact the clearing server.

The foregoing high level summary does not include several details and possible variations on method 1 of the invention including encryption/decryption, key generation, Token time stamps, and embodiments not requiring consumer software downloads. However it captures the key features discussed earlier namely:

1) On the consumer's computer, the Token resides in memory and is therefore difficult to access and modify.
2) The Token has a fixed life which is set so that given existing computational resources, encrypted information on the Token cannot be broken before the Token expires.
3) A new, unique Token is created for transactions with each new merchant.
4) Transactions with a merchant require only one request from the merchant to the clearing server after which subsequent transactions take place only between the merchant and the buyer.
5) Purchases produce immediate decreases in the displayed Token balance.
6) Various attempts at fraud such as using a stolen Token or trying to perform simultaneous transactions from two computers with the aim of spending more than the Token value all fail.

<u>Applications of my payment invention</u>
My payment inventions are ideal for making very small dollar purchases on the Internet (as little as 1¢) However myriad applications exist.

a. <u>Large Payments</u>: My invention can be used to pay for arbitrarily large transactions using the same underlying technology but with stronger Token encryption.

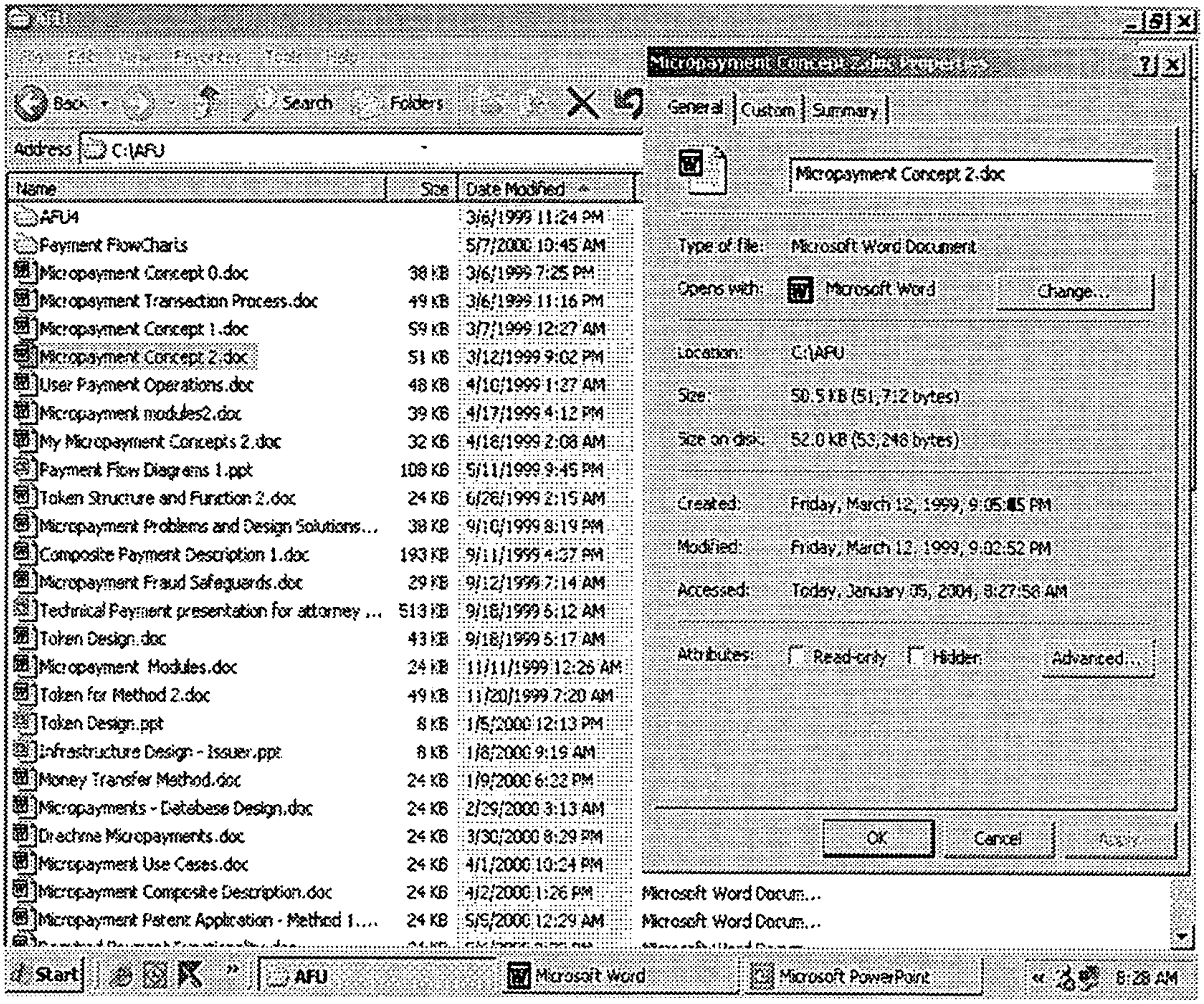
b. <u>Money Transfers</u>: My invention can be used to make auction payments very cheaply. The transferor goes to The Clearing server with his Token and requests transfer of $100 for example to a specified beneficiary. He selects a new login and new password to be provided by the beneficiary. The transferor's Token is then decreased by the $100 and a transfer number generated. The beneficiary from wherever he is located need only connect to the clearing server enter the login, password and transfer number to generate a new Token or credit his existing Token with the Transfer amount. The Token can then be converted to

cash either by being credited to a checking account or collected as cash from a disbursing location.

c. <u>Online purchase of foreign currency denominated products:</u> Buying a £1 item with a U.S. dollar Token will pop up a window that shows the price of the product in U.S. dollars at the going exchange rate. Making the purchase will decrease the buyer's Token by the dollar equivalent of the foreign currency price. In other words the buyers Token does not have to be converted into foreign currency in other to make a foreign currency purchase. This has obvious utility for reducing back office processing in currency trading of large amounts denominated in foreign currency. In other words, the buyer will authorize decrementing of a Token, if the dollar equivalent amount – that is the exchange rate -- is deemed acceptable.

d. <u>Cash redemption:</u> Any network connected to the Clearing server can be used to decrement a Token and issue cash in exactly the same way that a vendor decrements a Token and releases a product to a buyer in a sales transaction. The only difference is that the vendor instead of releasing a product to the buyer like a digital photograph, releases cash.

I predict that my payment invention will have several applications on the Internet. I will write out details on possible formats for the Token and describe how these provide security. I will also break down the various operations performed by the user into modules (like use cases) so that the actions of the user can be tied back to code. I will use this information to build a demo and how it works.


Back
Search
Folders
Address C:\AFU
Name
Size
Date Modified
AFU4 3/6/1999 11:24 PM
Payment FlowCharts 5/7/2000 10:45 AM
Micropayment Concept 0.doc 38 KB 3/6/1999 7:25 PM
Micropayment Transaction Process.doc 49 KB 3/6/1999 11:16 PM
Micropayment Concept 1.doc 59 KB 3/7/1999 12:27 AM
Micropayment Concept 2.doc 51 KB 3/12/1999 9:02 PM
User Payment Operations.doc 48 KB 4/10/1999 1:27 AM
Micropayment modules2.doc 39 KB 4/17/1999 4:12 PM
My Micropayment Concepts 2.doc 32 KB 4/18/1999 2:08 AM
Payment Flow Diagrams 1.ppt 108 KB 5/11/1999 9:45 PM
Token Structure and Function 2.doc 24 KB 6/26/1999 2:15 AM
Micropayment Problems and Design Solutions... 38 KB 9/10/1999 8:19 PM
Composite Payment Description 1.doc 193 KB 9/11/1999 4:27 PM
Micropayment Fraud Safeguards.doc 29 KB 9/12/1999 7:14 AM
Technical Payment presentation for attorney ... 513 KB 9/18/1999 5:12 AM
Token Design.doc 43 KB 9/18/1999 5:17 AM
Micropayment Modules.doc 24 KB 11/11/1999 12:26 AM
Token for Method 2.doc 49 KB 11/20/1999 7:20 AM
Token Design.ppt 8 KB 1/5/2000 12:13 PM
Infrastructure Design - Issuer.ppt 8 KB 1/8/2000 9:19 AM
Money Transfer Method.doc 24 KB 1/9/2000 6:23 PM
Micropayments - Database Design.doc 24 KB 2/29/2000 3:13 AM
Drachma Micropayments.doc 24 KB 3/30/2000 8:29 PM
Micropayment Use Cases.doc 24 KB 4/1/2000 10:24 PM
Micropayment Composite Description.doc 24 KB 4/2/2000 1:26 PM
Micropayment Patent Application - Method 1.... 24 KB 5/5/2000 12:29 AM
Microsoft Word Docum...
Microsoft Word Docum...
General
Custom
Summary
Micropayment Concept 2.doc
Type of file: Microsoft Word Document
Opens with: Microsoft Word
Change...
Location: C:\AFU
Size: 50.5 KB (51,712 bytes)
Size on disk: 52.0 KB (53,248 bytes)
Created: Friday, March 12, 1999, 9:05:45 PM
Modified: Friday, March 12, 1999, 9:02:52 PM
Accessed: Today, January 05, 2004, 8:27:58 AM
Attributes: Read-only Hidden
Advanced...
OK
Cancel
Apply
Start
AFU
Microsoft Word
Microsoft PowerPoint
8:28 AM

The Micropayment Problem. My Insights and Solutions

All payment technologies are of 2 types: notational systems and value systems.

A. Notational systems are best represented by payment by check and involve 3 parties: A consumer, a merchant and a bank. A consumer pays a merchant by check, the merchant deposits the check at his bank. His bank presents the check to the consumer's bank. The consumer's bank debits the consumer's account, credits the merchant's account--that is the check is cleared--and informs the merchant who can then release the goods. The notational system has a lot of overhead and is therefore unsuitable for micropayments. The time overhead is so great that even for brick and mortar transactions the merchant typically collects customer identifying information and releases the sold goods before the consumer's check clears.

B. Value systems use a digitally encrypted message, called a token, which holds monetary value. The token, is presented by the consumer to the merchant to make a purchase. The merchant ascertains that the token is authentic, usually by decrypting the encrypted signature of the issuer, and can thereafter debit the token directly for a sale without having to contact a third party, (the bank issuer). The overhead costs are lower since no third party has to be consulted and this makes value systems the method of choice for micropayments.

<u>Problems with Value systems</u>. Value systems require that a token representing an encrypted message hold money value which can be transferred to a merchant without the need to contact the issuer of the token. This presents several problems:

1) <u>Double spending</u>. Since the token is digital, the consumer can, in theory, make numberless copies and present each copy to a different merchant. Each copy is a perfect replica of the original and will pass any authentication tests. In other words, the consumer can spend the same token two or more times. This is the "double spending problem". Several solutions are plausible.
    a. The merchant could contact the token issuer for every transaction to confirm that the token received has not been "spent" before. This re-creates the overhead problem of the Notational system, since the merchant must contact the issuer for every transaction. In addition, the issuer must have continuously updated records of all tokens outstanding (issued to consumer) and all tokens redeemed (spent by consumer and returned to the Issuer by the merchant) . The storage requirements are extremely large as is the computational load on the issuer. A token for example which was last spent 9 months ago must be retrieved in response to a merchant inquiry. If the token was last spent 5 years ago, the time taken for retrieval becomes unacceptably large. If the issuer is processing several requests, the issuer will be unable to handle very many transactions. Preventing double spending through confirmation with the token issuer also assumes that the merchant immediately after he receives an authentic token from a sale, instantly deposits it with the issuer. If there is a time lag between receiving a genuine payment and depositing it with the issuer, then double spending can still take place. The fraudulent consumer simply makes a purchase at a merchant and before that merchant deposits the token, makes a second purchase at another merchant with a copy of the token. Since the first merchant has not deposited the original token with the issuer, an inquiry to the issuer by the second merchant will confirm that the token is real, was issued and has not been redeemed.

    b. A more common solution is to the double spending problem is to introduce a hardware device that would hold the token and make it inaccessible to the consumer and therefore impossibie to copy. Transactions would require that the merchant access the token

within this hardware device, authenticate and then debit the token to conclude a sale. This requires that both the consumer and the merchant use a special device into which the token would be downloaded. The token will therefore be inaccessible to the consumer and therefore cannot be copied. This solves the double spending problem but means that micropayment transactions, would require every participant, consumer or merchant to have a smart card, which, in turn, makes adoption insuperably difficult. All hardware solutions imply a very high adoption hurdle and an exponentially high likelihood of commercial failure.

2) The Change problem. Assume that double spending was somehow solved in software, other problems remain. Imagine a scenario where a consumer presents a $10 token and wishes to purchase a 5¢ item. The merchant needs to "create" a new token with value $9.95 and ensure that it has all the security and acceptability features of the original $10 token. So to make change each merchant issues new cash. All participants in such a system must agree a priori to issue currency and accept each others currencies. Some solutions to the change problem require the consumer to carry multiple tokens (of one issuer) in several denominations. It is easy to derive the minimum number of tokens required in order to make a payment of any amount below $1 for example. The problems is that after any purchase, the remaining number of tokens can no longer be used to pay for any amount and the consumer must replace the depleted tokens after each transaction. A consumer with 5 pennies, 1 nickel, 4 dimes and 2 quarters can make payment for any article that costs a dollar or less without requiring change. The consumer may for example be able to buy a 49¢ item once but thereafter will be unable to purchase a 38¢ item without replacing his depleted dimes. Token replacement, after each transaction increases overhead and means again that small transactions become impractical.

3) The Money Value problem For the token to be used in purchases, it must have money value attached to it. The options are for the consumer to "pay" with conventionally accepted currency in order to create a token that is backed by money or for a consumer or issuer to create "currency" not backed by money but which is redeemable for money after it is spent. In the latter case, a mechanism must exist to charge the consumer and credit the merchant post-transaction. It also requires all transacting parties to believe that the "currency" they create can be redeemed for cash. This creates problems of convertibility, and requires the issuer to be a "Central Bank". This makes adoption very difficult.

4) The Security problem Providing security for small transactions has proven intractable for many operators. Most payment systems use Private/Public Key (PPK) encryption. This is a technique where the encryption key and the decryption keys are different. One party, usually the issuer, distributes his public key widely and encrypts tokens with his private key (which no other party knows). The public key can then be used to decrypt any token written with the issuers private key. The problem is that PPK is computational intensive and requires each merchant to possess a super computer in order to process a moderate number of transactions. People have tried to reduce the computation load by not encrypting the entire token but encrypting instead a numeric value obtained by running the token through a specific algorithm. The recipient of the token, decrypts the encrypted hash value and then runs the token through the same algorithm. It the value obtained matches the value decrypted, the token is assume to be authentic. Using hashes as opposed to encrypting the whole token only moderately reduces the computational load. Dropping the encryption requirement compromises security. In fact some, micropayment methods require the consumers and merchants to be willing to lose money on their payment transactions.

I believe that the problems encountered in solving the Micropayment problem are very similar to the problems encountered when man first tried to fly. The idea was to imitate birds by obtaining wings that flap. Flight was only perfected when that idea was abandoned and instead the Wright Brothers pursued the idea of staying airborne, which they realized could be done without flapping wings. In the same way, most digital payment systems are trying to duplicate the transactional operations of currency: You give a merchant cash – the palpable hard coins or notes that you feel in your wallet -- and the merchant gives you the item you purchased with any change due. This effort to replicate the operations performed in a cash transaction are both difficult and unnecessary in a digital medium. Digital money can and should perform the exchange and storage functions of money without replicating the look, feel or operational steps performed with physical cash.

My solution reduces overhead, enables scalability and ensures security without raising transaction cost. I describe the conceptual basis of my digital payment method below.

1) CORE CONCEPT
    a. Have a single token that is really a pointer to information that resides elsewhere such as on a clearing server. The token simply holds at a minimum a unique number and PIN. The token number and PIN is the minimum required to retrieve the actual records for that token which reside on a remote server. These records include the balance on the token, what transactions were performed on the token, by whom the transactions were performed and whether the token is currently "checked out" i.e. residing with a merchant , or if it is held by the clearing server. For each transaction, a part of the token record moves to the transaction location. I have 2 solutions: One where the token record moves to the merchant and another completely different solution where it moves to the consumer. I will file patents for both solutions.

    b. In either, case debits and credits occur at the transaction location between the merchant and the consumer. When the transactions are completed the record for that token is returned to the clearing server.

2) TOKEN MOVES TO MERCHANT
    a. In the case where, the token is sent to the merchant, the consumer enters the token number and PIN in order to complete a purchase. The merchant forwards this number and PIN to the central which first authenticates the merchant and then sends him the token record with authority to decrement or increment the balance on the token and record the transactions associated with each increment and decrement. Once the token arrives at the merchant, (i.e. merchant merchant's computer), he can debit or credit it multiple times without the need to contact the clearing server. The token once it is downloaded is "local" and so there is very little overhead cost in processing subsequent transactions, small or large after the initial download to the merchant. The merchant has to have registered beforehand with the clearing server and obtained software which does processing on the merchant's server.

    b. The clearing server simply authenticates the requesting merchant, identifies the record associated with the token number and PIN, updates the records associated with that token if needed (the token record may currently be "checked out" i.e., residing with another merchant and will have to be retrieved), and then downloads the token record with a key to the requesting merchant. The clearing server is not burdened with processing actual transactions and can therefore handle a very, very large number of token requests. Since actual processing occurs at the merchant's computer this

represents a distributed solution and means that transaction processing can scale i.e., that is the number of transactions processed can be increased almost without limit.

3) TOKEN MOVES TO CONSUMER.

a. In the case where processing is done on the consumers device, the same advantages described above also accrue. The processing is distributed, the transactions scale.

b. The consumer has to first purchase and download a token along with software that manages the token on the consumers computer including: incrementing and decrementing the token when a key is received from a merchant, keeping the token in RAM, destroying the token when it expires etc. The consumer downloads this software with the token from the clearing server when first registering.

c. The consumer presents this token to the merchant in order to perform a transaction. The consumer may be required to enter a PIN number for additional security.

d. The merchant forwards the consumer's token to the clearing server which authenticates and gives the merchant a key that authorizes the merchant to modify (decrement or increment) the value of the token on the consumer's computer. The modification is performed when the merchant presents a request for a decrement or increment along with a key to the software on the consumer's computer. It is actually the consumer's software which does the decrement/increment of the token, although the merchant's software also records the amount of the decrement and the specific token involved. It is the merchant's token data which is used to update the token.

e. The consumer can then perform several transactions with that merchant who can decrement the token immediately without having to contact the clearing server. The merchant keeps a record of the transactions performed on the token and uploads these transactions to the clearing server periodically or in response to a poll from the clearing server. Once a token record is uploaded no further transactions can be performed on the token, unless the merchant makes another request for the token and update key along with the upload.

f. Consumer transactions with a merchant a described as "sessions". A session is an uninterrupted period of transactions with ONE merchant, without any intervening transactions with another merchant and before expiration of the token. During a session, the merchant has the key and the merchant can continue modifying the token without contacting the clearing server.

g. If the consumer should move from the first merchant to a second merchant to transact, he again submits his token, to the second merchant, who again forwards this token to the clearing server. This time the clearing server notices that a token key was given to the first merchant who has not yet sent an upload. It polls the first merchant obtains token updates, recalculates the balance on the token, creates a new token and then downloads this new token with 2 keys (they could really be one key) an update key and an overwrite key, to the second merchant. This is a second unique "session".

h. This second merchant then overwrites the consumer's old token with the new one and can then begin decrementing or incrementing the token with the second key. The primary advantage of delivering the token to the consumer is that it can be used for transactions offline as the consumer does not require a connection to the Internet or a

network. All that is required is connection to the merchant's computer which can be achieved either by a direct physical connection, infra-red, blue tooth, short range wireless, etc. Since the token holds a running balance of transactions performed, the consumer also obtains a real time balance of transactions. The problems are that since the token itself is the primary authorization required to receive a key, it must have security features which make it difficult to duplicate.

i. The security features are:

   i. The token resides in RAM on the consumer's computer and disappears when the consumer's computer is switched off. The consumer has to contact the clearing server to re-obtain the token in this case. This feature can be turned off for portable handheld devices, so that the consumer can turn off the device turn it back on and still have the token. Since the token is modified by the merchant as the consumer shops, the consumer can use the device to shop offline, while still obtaining a running transaction balance. The merchant's however have to be connected to the clearing server. The capacity to obtain a real-time balance while transacting is not as far as I know implemented in any payment system apart from cash.

   ii. The token has a short life and expires after a few hours. This can be modified but is a safety precaution. If the token expires while on the consumer's computer it is removed from RAM by the consumer's software. If it expires while the consumer is transacting with a merchant, the merchant uploads all the records and requests a new token and a new key.

   iii. Every time the token is re-obtained it has a different number known only by the clearing server as well as modified fields, including date and expiration. Each token is therefore different. The token fields are stored on the clearing server. A token presented by a merchant has to match the stored fields which makes duplication impossible.

   iv. Although the token itself is likened to money, it is really an authentication key which the merchant forwards to the clearing server.

4) SCALING THE CLEARING SERVER

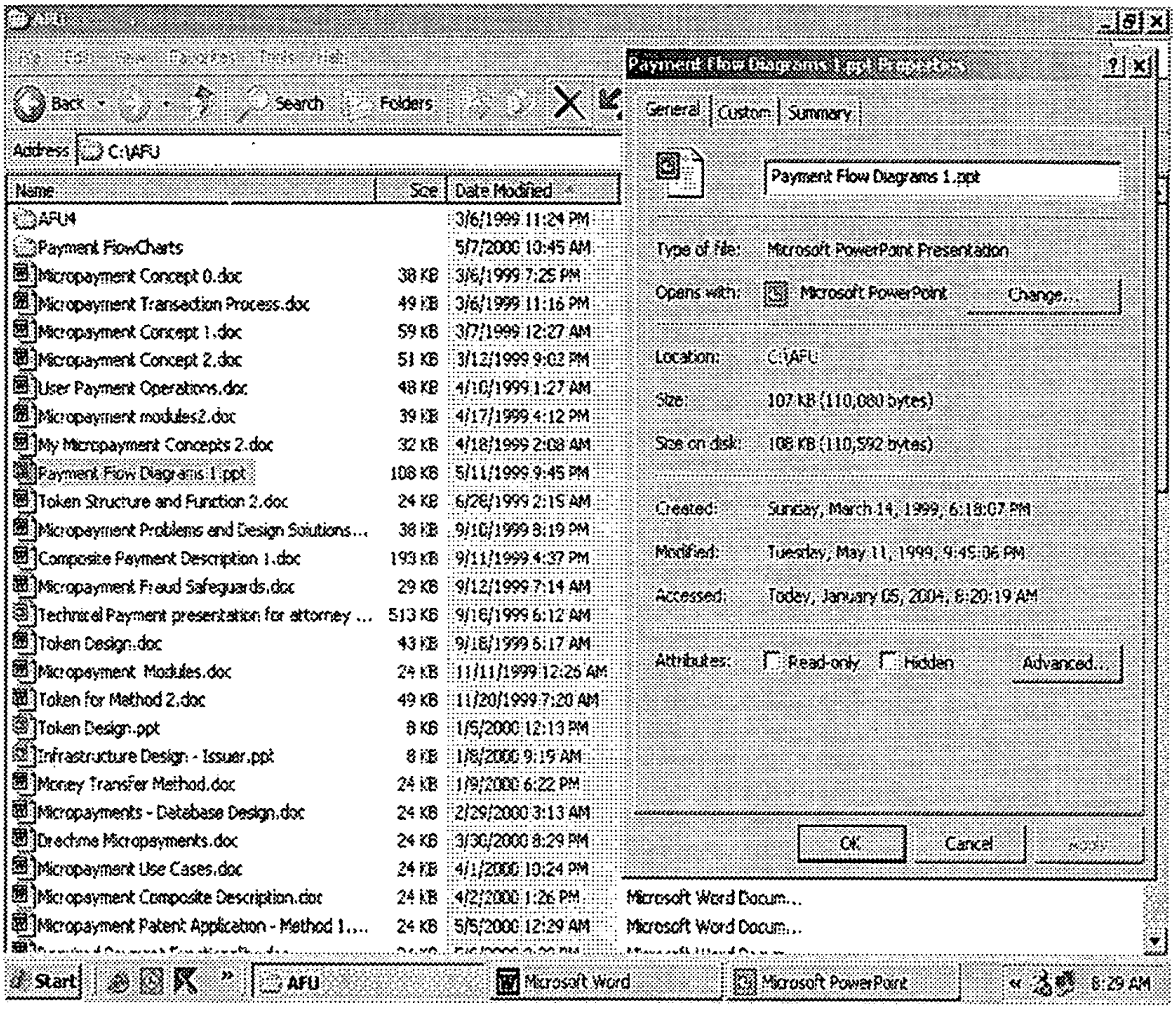
   a. There will be several clearing servers that hold token records and there are very many schemes to divvy up which servers hold which token records. These include breaking the tokens into ranges with different servers holding different records. The idea is for each clearing server to generate a token number in a way which uniquely identifies that token as belonging to that clearing server. In other words, there will be a number of token clearing servers with responsibility for token records in certain number ranges. This is similar to but not identical with, and certainly not the same as, the domain name system (DNS).

   b. The token Number could for example be a 17 digit number (It could also be alpha numeric to increase the range). This gives a set of 10^{17} possible numbers. Each clearing server will have a range of numbers within this set for which it is responsible for example, the first clearing server could own the range of 17 digit numbers beginning with 1and 2. Another server would own the range of 17 digit numbers beginning with 3

and 4 and so on. Each clearing server would in turn have 0 or more subordinate servers who would own records within the number subset handled by their parent clearing server. These subordinate servers will therefore only create tokens with numbers within the number subset allocated to the clearing server. The clearing server will determine the portions of its number subset to be handled by the various subordinate servers.

c. A concrete example. A consumer submits to a merchant the token number 1234 5678 9012 34567 and PIN. The clearing server that receives this request serve only token numbers beginning with 3 and 4, so it forwards the token request to the clearing server responsible for numbers 1 and 2. This server receives the request and then checks its internal tables, to identifies the subordinate server that handles token number 1234 and forwards this request to the server. The subordinate server may in turn process the request or forward it to its own subordinate servers to whom it delegates portions of its own number subset, perhaps a server handling tokens with numbers in the range 1234 5xxx to 1234 6xxx and so on. The server which actually holds the token record then forwards it either directly to the merchant or to the initial requesting clearing server or to the server where the merchant's registration record resides, for onward delivery to the merchant. The merchant performs transactions on the token record and then returns it to its clearing server either as part of its periodic upload or in response to a poll. I'll expand on this distributed clearing server processing in a separate document. The point is that it means that my payment method can scale to handle billions of transactions, and will not be limited by geography.

5) The download of the token to the merchant requires that the merchant be registered beforehand with the clearing server and that the merchant receive software from the clearing server. This merchant will present a key which was received at registration along with its request for a token record. Initially "handshake" between the clearing server and the merchant can be performed with this key and will make merchant authentication much faster as the clearing server does not have to first negotiate and agree on a unique key as required in Secure Socket Layer (SSL) transactions.


AFU
Payment Flow Diagrams 1.ppt Properties
Back
Search
Folders
Address C:\AFU
Name
Size
Date Modified
AFU4 3/6/1999 11:24 PM
Payment FlowCharts 5/7/2000 10:45 AM
Micropayment Concept 0.doc 38 KB 3/6/1999 7:25 PM
Micropayment Transaction Process.doc 49 KB 3/6/1999 11:16 PM
Micropayment Concept 1.doc 59 KB 3/7/1999 12:27 AM
Micropayment Concept 2.doc 51 KB 3/12/1999 9:03 PM
User Payment Operations.doc 48 KB 4/10/1999 1:27 AM
Micropayment modules2.doc 39 KB 4/17/1999 4:12 PM
My Micropayment Concepts 2.doc 32 KB 4/18/1999 2:08 AM
Payment Flow Diagrams 1.ppt 108 KB 5/11/1999 9:45 PM
Token Structure and Function 2.doc 24 KB 6/26/1999 2:15 AM
Micropayment Problems and Design Solutions... 38 KB 9/10/1999 8:19 PM
Composite Payment Description 1.doc 193 KB 9/11/1999 4:37 PM
Micropayment Fraud Safeguards.doc 29 KB 9/12/1999 7:14 AM
Technical Payment presentation for attorney ... 513 KB 9/16/1999 6:12 AM
Token Design.doc 43 KB 9/18/1999 5:17 AM
Micropayment Modules.doc 24 KB 11/11/1999 12:26 AM
Token for Method 2.doc 49 KB 11/20/1999 7:20 AM
Token Design.ppt 8 KB 1/5/2000 12:13 PM
Infrastructure Design - Issuer.ppt 8 KB 1/8/2000 9:19 AM
Money Transfer Method.doc 24 KB 1/9/2000 6:22 PM
Micropayments - Database Design.doc 24 KB 2/29/2000 3:13 AM
Drechme Micropayments.doc 24 KB 3/30/2000 8:29 PM
Micropayment Use Cases.doc 24 KB 4/1/2000 10:24 PM
Micropayment Composite Description.doc 24 KB 4/2/2000 1:26 PM
Micropayment Patent Application - Method 1.... 24 KB 5/5/2000 12:29 AM
Microsoft Word Docum...
Microsoft Word Docum...
General
Custom
Summary
Payment Flow Diagrams 1.ppt
Type of file: Microsoft PowerPoint Presentation
Opens with: Microsoft PowerPoint
Change...
Location: C:\AFU
Size: 107 KB (110,080 bytes)
Size on disk: 108 KB (110,592 bytes)
Created: Sunday, March 14, 1999, 6:18:07 PM
Modified: Tuesday, May 11, 1999, 9:45:06 PM
Accessed: Today, January 05, 2004, 8:20:19 AM
Attributes: Read-only Hidden
Advanced...
OK
Cancel
Start
AFU
Microsoft Word
Microsoft PowerPoint
8:29 AM

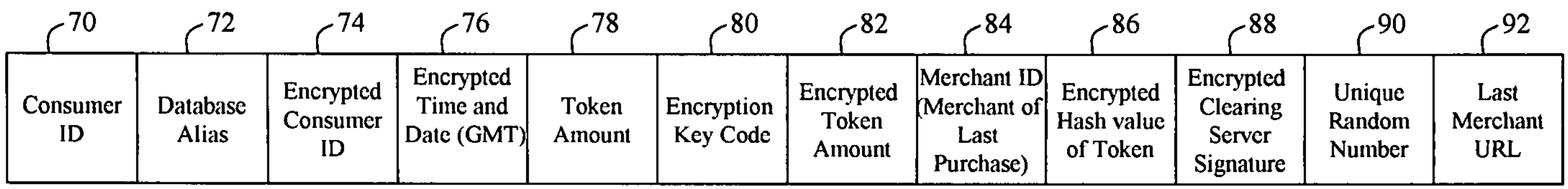

70
Consumer ID
72
Database Alias
74
Encrypted Consumer ID
76
Encrypted Time and Date (GMT)
78
Token Amount
80
Encryption Key Code
82
Encrypted Token Amount
84
Merchant ID (Merchant of Last Purchase)
86
Encrypted Hash value of Token
88
Encrypted Clearing Server Signature
90
Unique Random Number
92
Last Merchant URL

Figure 1

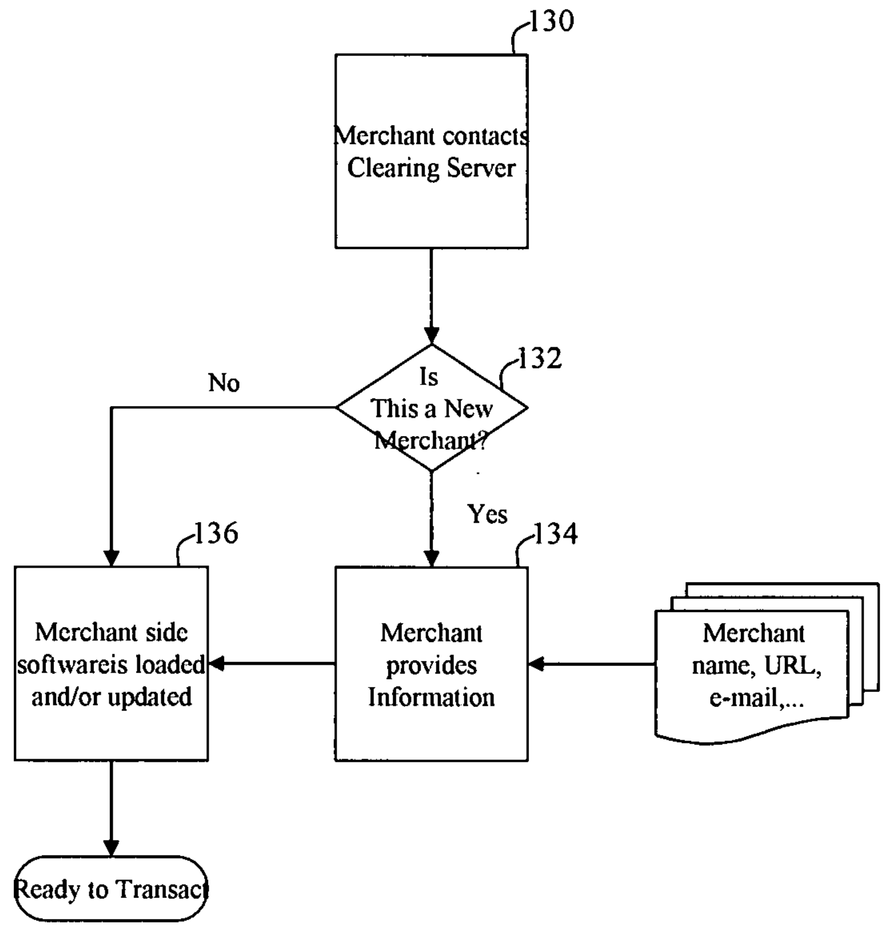

130
Merchant contacts Clearing Server
132
Is This a New Merchant?
No
Yes
136
134
Merchant side softwareis loaded and/or updated
Merchant provides Information
Merchant name, URL, e-mail,...
Ready to Transact

Figure 2

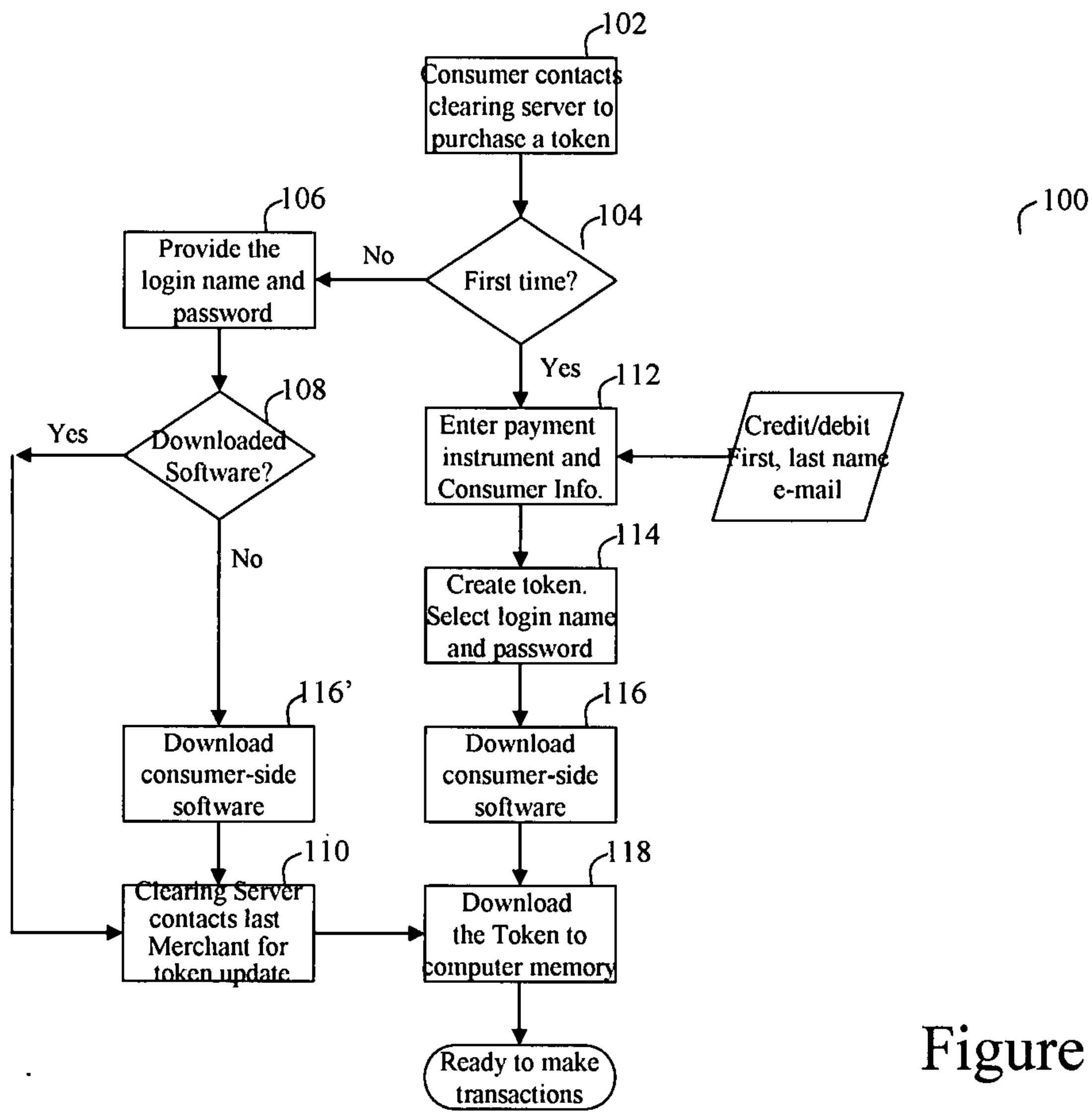

102
Consumer contacts clearing server to purchase a token
104
First time?
No
Yes
106
Provide the login name and password
108
Downloaded Software?
Yes
No
112
Enter payment instrument and Consumer Info.
Credit/debit First, last name e-mail
114
Create token. Select login name and password
116'
Download consumer-side software
116
Download consumer-side software
110
Clearing Server contacts last Merchant for token update
118
Download the Token to computer memory
Ready to make transactions
100

Figure 3

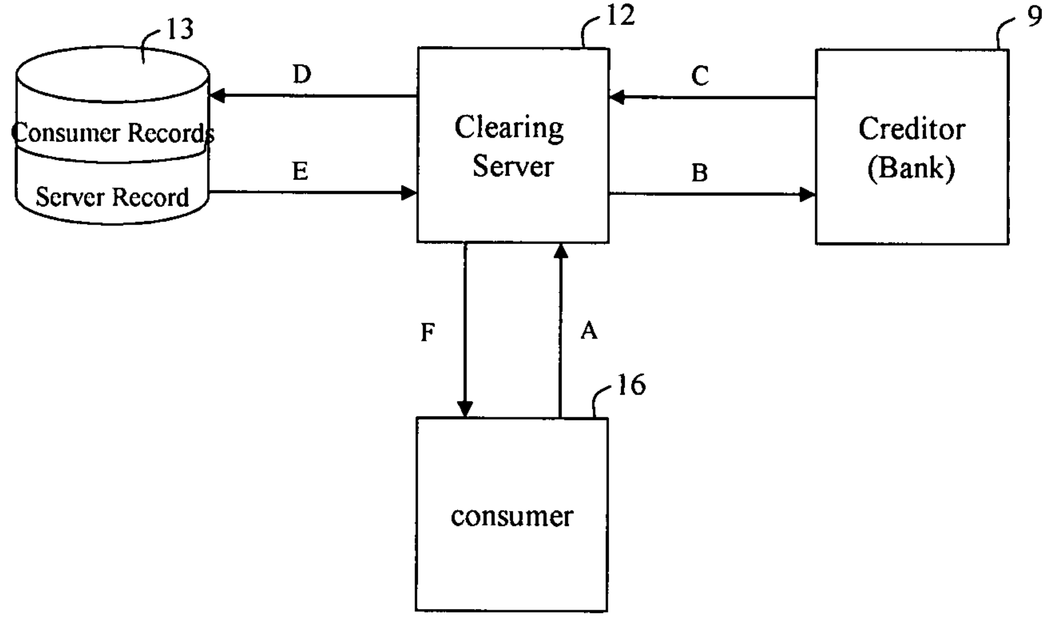

13
Consumer Records
Server Record
D
E
12
Clearing Server
C
B
9
Creditor (Bank)
F
A
16
consumer

Figure 4

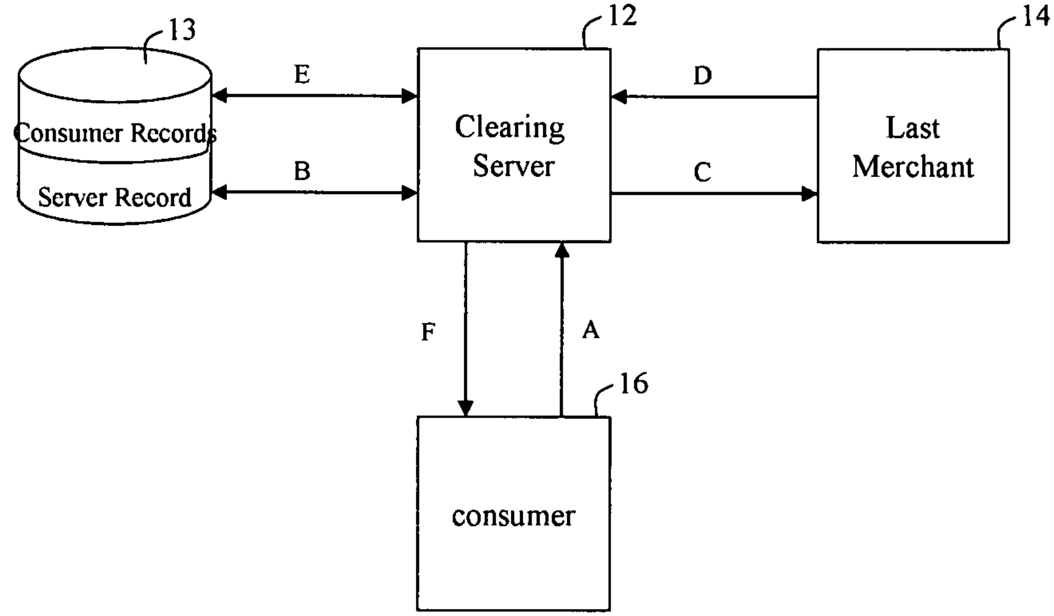

13
Consumer Records
Server Record
E
B
12
Clearing
Server
D
C
14
Last
Merchant
F
A
16
consumer

Figure 5

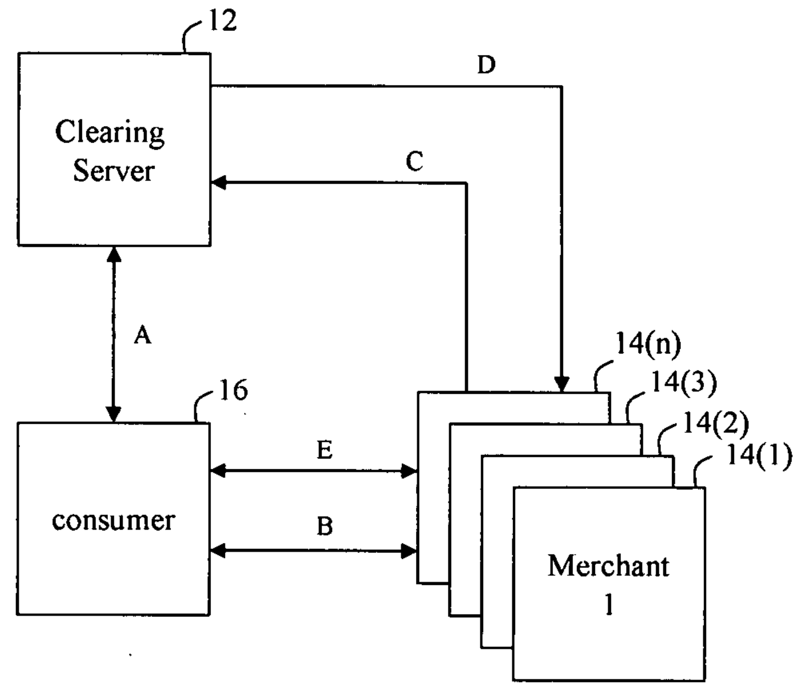
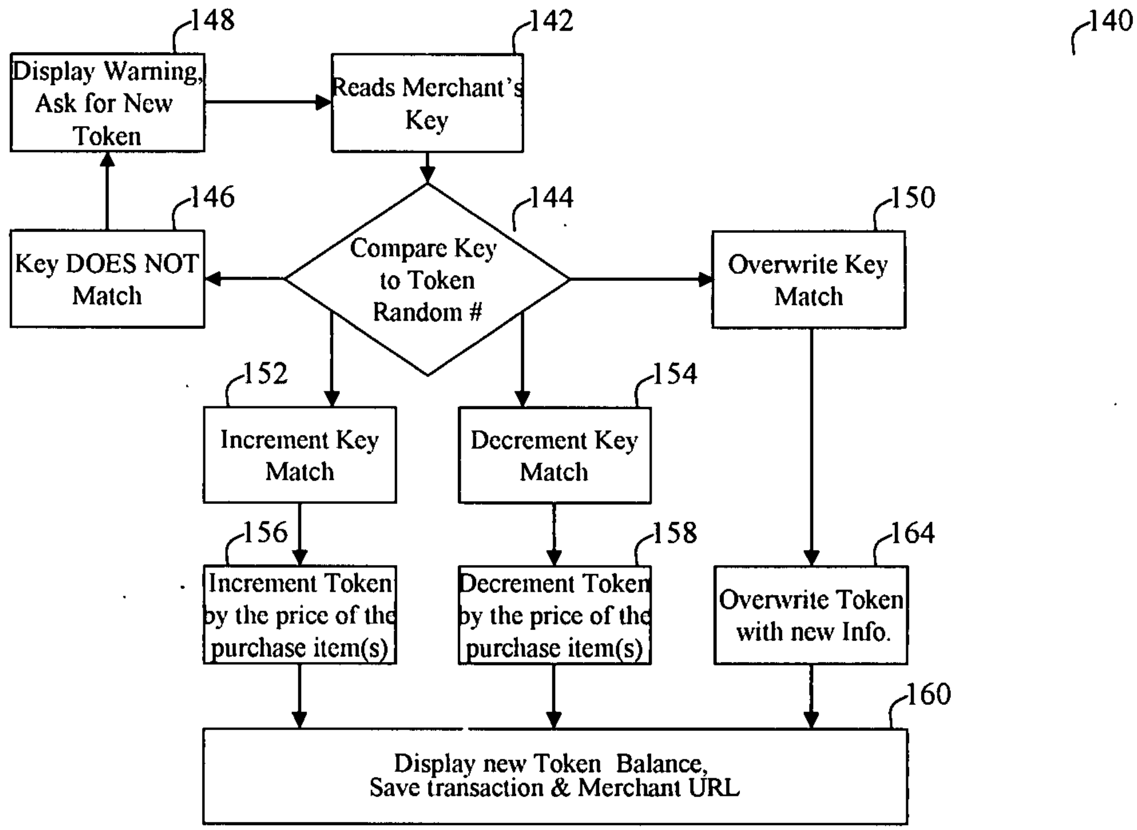

12
Clearing
Server
D
C
A
16
consumer
E
B
14(n)
14(3)
14(2)
14(1)
Merchant
1

Figure 6



148
Display Warning, Ask for New Token
142
Reads Merchant's Key
140
146
Key DOES NOT Match
144
Compare Key to Token Random #
150
Overwrite Key Match
152
Increment Key Match
154
Decrement Key Match
156
Increment Token by the price of the purchase item(s)
158
Decrement Token by the price of the purchase item(s)
164
Overwrite Token with new Info.
160
Display new Token Balance, Save transaction & Merchant URL

Figure 7

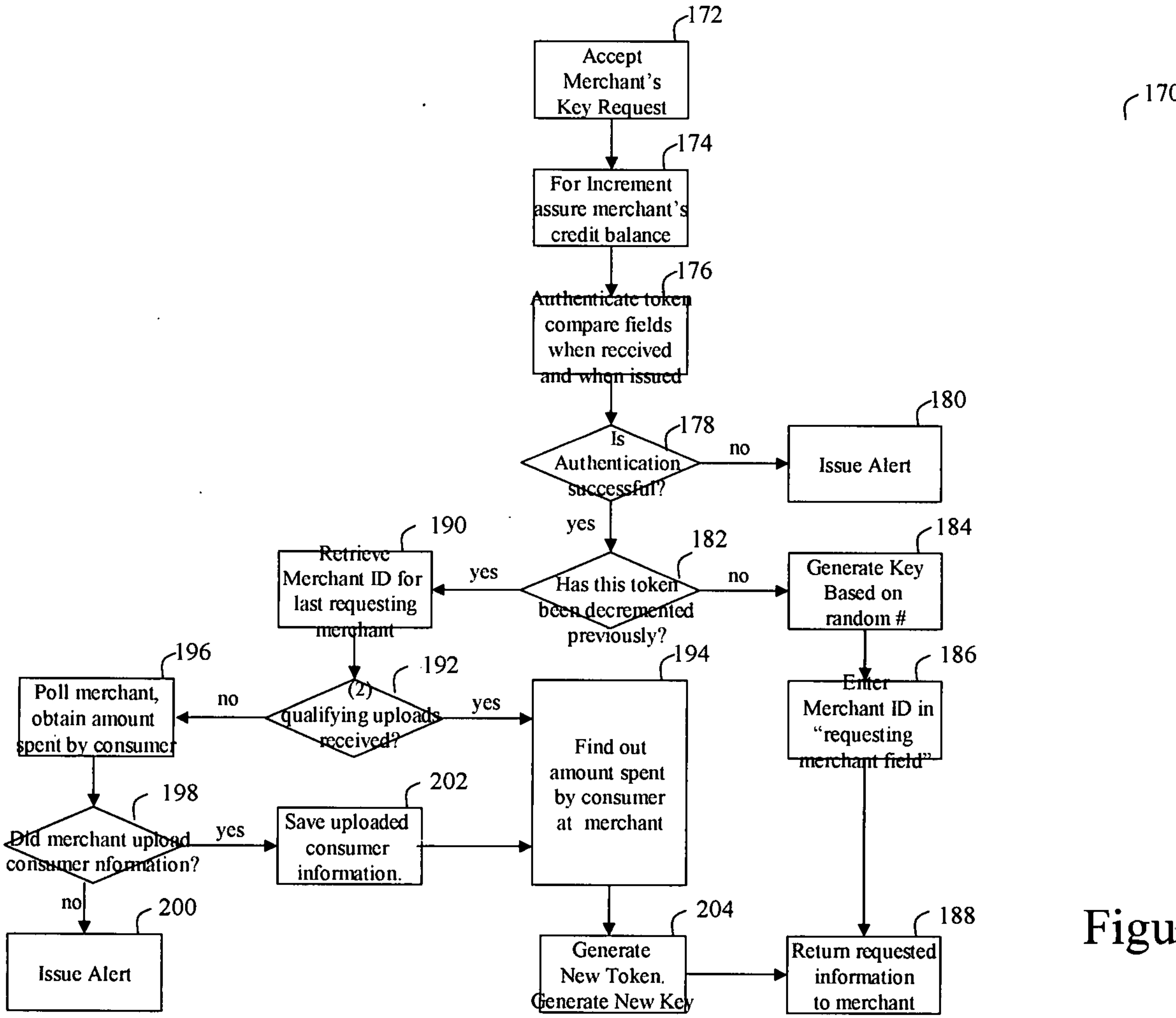

170
172
Accept Merchant's Key Request
174
For Increment assure merchant's credit balance
176
Authenticate token compare fields when received and when issued
178
Is Authentication successful?
no
180
Issue Alert
yes
182
Has this token been decremented previously?
no
184
Generate Key Based on random #
186
Enter Merchant ID in "requesting merchant field"
188
Return requested information to merchant
yes
190
Retrieve Merchant ID for last requesting merchant
192
(2) qualifying uploads received?
yes
194
Find out amount spent by consumer at merchant
204
Generate New Token. Generate New Key
no
196
Poll merchant, obtain amount spent by consumer
198
Did merchant upload consumer nformation?
yes
202
Save uploaded consumer information.
no
200
Issue Alert

Figure 8

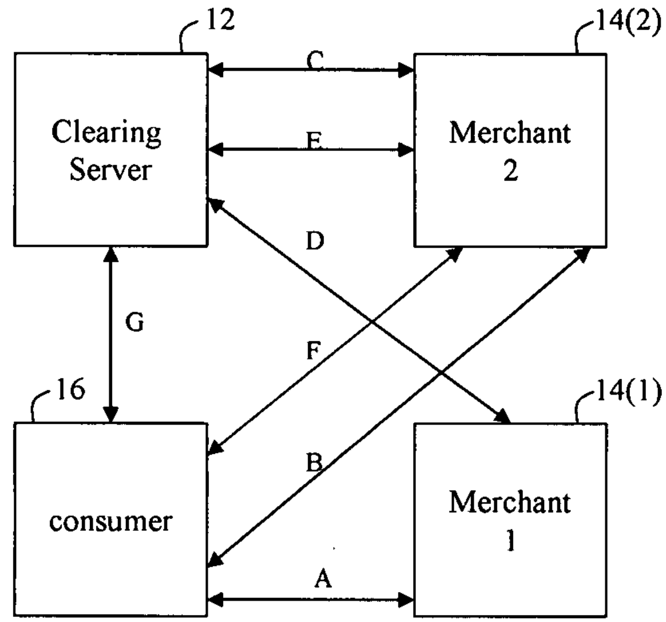

12
14(2)
Clearing Server
C
E
Merchant 2
D
G
F
16
14(1)
B
consumer
Merchant 1
A

Figure 9

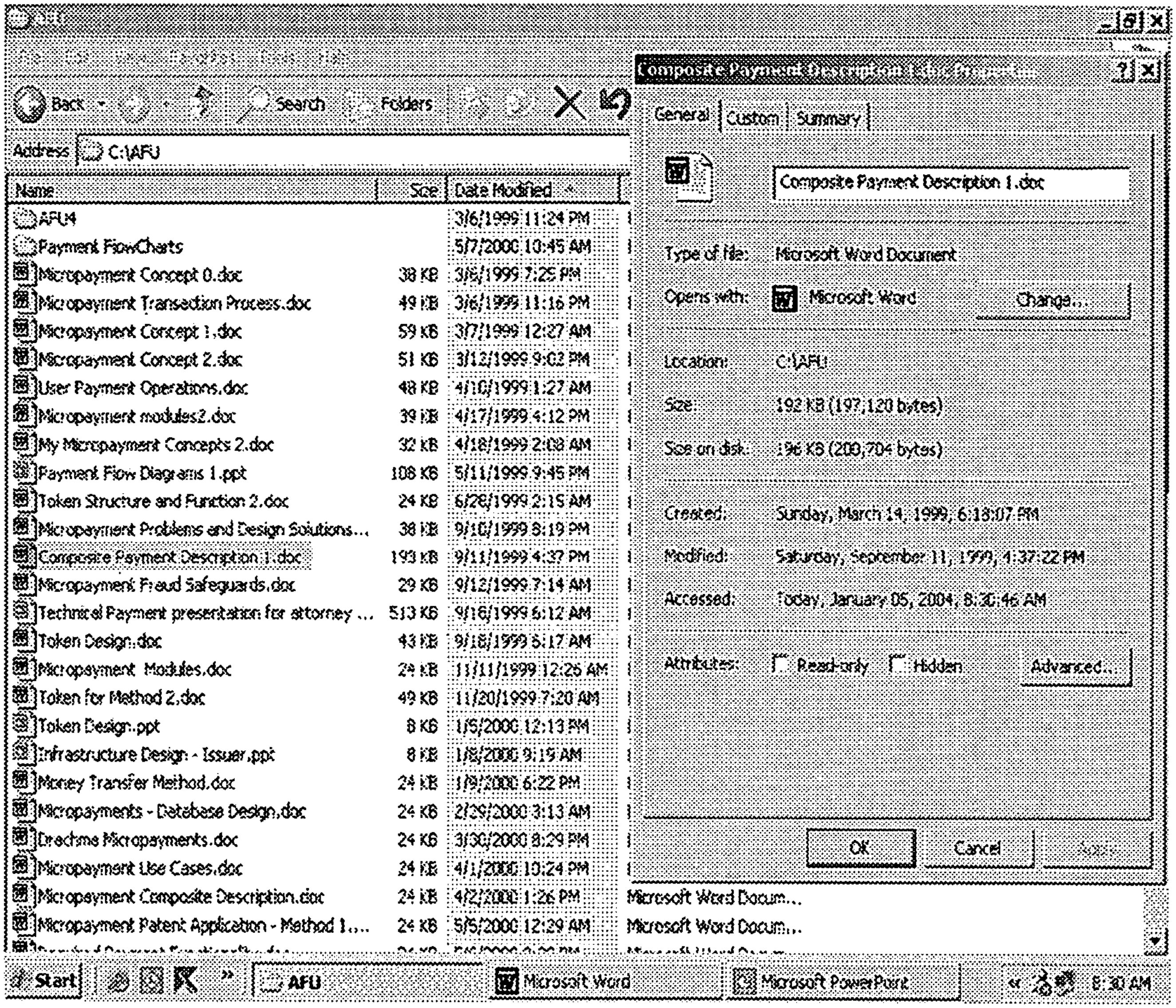

Composite Payment Description 1.doc Propert...
Back
Search
Folders
Address C:\AFU
Name
Size
Date Modified
AFU4 3/6/1999 11:24 PM
Payment FlowCharts 5/7/2000 10:45 AM
Micropayment Concept 0.doc 38 KB 3/6/1999 7:25 PM
Micropayment Transaction Process.doc 49 KB 3/6/1999 11:16 PM
Micropayment Concept 1.doc 59 KB 3/7/1999 12:27 AM
Micropayment Concept 2.doc 51 KB 3/12/1999 9:02 PM
User Payment Operations.doc 48 KB 4/10/1999 1:27 AM
Micropayment modules2.doc 39 KB 4/17/1999 4:12 PM
My Micropayment Concepts 2.doc 32 KB 4/18/1999 2:08 AM
Payment Flow Diagrams 1.ppt 108 KB 5/11/1999 9:45 PM
Token Structure and Function 2.doc 24 KB 6/28/1999 2:15 AM
Micropayment Problems and Design Solutions... 38 KB 9/10/1999 8:19 PM
Composite Payment Description 1.doc 193 KB 9/11/1999 4:37 PM
Micropayment Fraud Safeguards.doc 29 KB 9/12/1999 7:14 AM
Technical Payment presentation for attorney ... 513 KB 9/16/1999 6:12 AM
Token Design.doc 43 KB 9/18/1999 5:17 AM
Micropayment Modules.doc 24 KB 11/11/1999 12:26 AM
Token for Method 2.doc 49 KB 11/20/1999 7:20 AM
Token Design.ppt 8 KB 1/5/2000 12:13 PM
Infrastructure Design - Issuer.ppt 8 KB 1/8/2000 9:19 AM
Money Transfer Method.doc 24 KB 1/9/2000 6:22 PM
Micropayments - Database Design.doc 24 KB 2/29/2000 3:13 AM
Drachma Micropayments.doc 24 KB 3/30/2000 8:29 PM
Micropayment Use Cases.doc 24 KB 4/1/2000 10:24 PM
Micropayment Composite Description.doc 24 KB 4/2/2000 1:26 PM Microsoft Word Docum...
Micropayment Patent Application - Method 1.... 24 KB 5/5/2000 12:29 AM Microsoft Word Docum...
General
Custom
Summary
Composite Payment Description 1.doc
Type of file: Microsoft Word Document
Opens with: Microsoft Word
Change...
Location: C:\AFU
Size: 192 KB (197,120 bytes)
Size on disk: 196 KB (200,704 bytes)
Created: Sunday, March 14, 1999, 6:13:07 PM
Modified: Saturday, September 11, 1999, 4:37:22 PM
Accessed: Today, January 05, 2004, 8:30:46 AM
Attributes: Read-only Hidden
Advanced...
OK
Cancel
Apply
Start
AFU
Microsoft Word
Microsoft PowerPoint
8:30 AM

Table of Contents

Section I (Required Information)

Consumer Information

To purchase the token the consumer using a browsing program enters information such as credit, debit, or prepaid card reference numbers into a secure form page on the Clearing server. The consumer is asked to select a password and a login id. The accepted information is stored in the Clearing server database as a unique consumer record and is used to create and update the token being purchased. Using three tables permits splitting the information into:

a. Static consumer information records comprising the following fields: a consumer ID; a first, middle, and last name; e-mail; city and country; database alias; software downloaded (Y/N); and software version. The software referenced in the last two fields will be discussed below.

b. Consumer secret information records comprising the following fields: a consumer ID; a payment instrument name and type, e.g., visa, master card, debit card, prepaid card, other; a name of the owner of the instrument; an instrument number and expiration; a login name; an encrypted password; an encrypted password reminder; the dollar amount purchased or face value of the token at purchase; date and time; and authorization information.

c. Changing consumer information records change each time the Clearing server updates the token, they comprise the following fields: a consumer ID; a first random number used to generate an update key; a second random number used to generate an overwrite key; a token balance; a last vendor purchased from; the second to last vendor purchased from; a requesting vendor; the request time; and a token hash value.

Where the consumer wishes to remain anonymous, the prepaid card records having a schema similar to the consumer records may be used. The main exception to creation of the consumer record is that the record has been pre-created in the Clearing server database and has as its consumer ID a number assigned to the prepaid card. The prepaid card record has numbers in the fields for first, middle, and last names, e-mail, and city and country names. Instead of entering his or her name, the consumer enters the prepaid card number and is prompted to enter the login name and password. This allows the consumer to retrieve the tokens purchased with the prepaid card.

Vendor Information

Using browsing programs vendors register with the Clearing server by providing information such as the name, e-mail, uniform resource locator (URL) or the TCP/IP address associated with the vendor's computing device or Internet website, creditor, and creditor account number are entered. This information is used to authenticate the vendor and in order to download software from the Clearing server. The vendor record, which is stored in the Clearing server database, holds static information about the vendor, detailed transaction information for each consumer and aggregate transaction information. The aggregate transaction information with the same schema is created by the vendor in a vendor database. <u>This vendor database is accessed when the Clearing server polls the vendor to retrieve the latest transactional information. The Clearing server may thus obtain the total amount spent by the consumer at the vendor website and then update the token to reflect this amount.</u>

a. Static vendor information records comprising the following fields: a vendor ID; name; URL; e-mail; address; login name; password (encrypted); a type of vendor business; date created; a software version; an encryption code; a vendor creditor's name; a creditor account number; a creditor routing number; and an authorized flag (y/n).

b. Aggregate vendor and consumer information records include the following fields: a vendor id; a consumer id (foreign key); a sum of consumer purchases; a total vendor credit; a total vendor debit; a net vendor credit; a date; and a time.

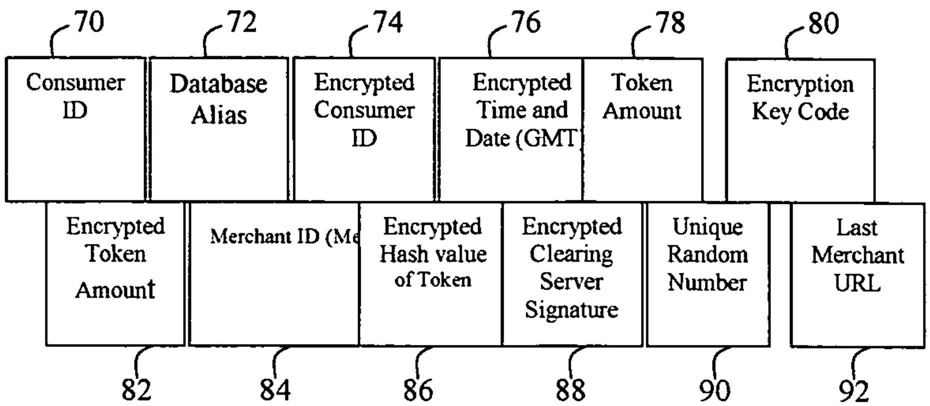
c. Vendor transaction information records comprising the following fields: a date; a vendor id; a consumer id; a product or service purchased; a consumer debit (token decrement); a consumer credit (token increment); and time.

Clearing Server Information

The Clearing server information records comprise the following fields: a Clearing server signature; an encryption key code; a message to be authenticated (unencrypted); a message to be authenticated (encrypted). These fields are extracted in creating a token or in creating a message on the Clearing server's "authentication page" to be used for authenticating of the vendors and consumers. The Clearing server signature field is extracted from the Clearing server information record, encrypted, and written to the token. The vendor maintains a digital copy of the unencrypted signature. The encryption key code is an alias for the key and type of encryption used by the Clearing server. This encryption key code, as will be described below in detail, is written to the token. The vendor using the old encryption key cannot decrypt information written with a new encryption key. Periodic updates of the vendor's encryption key enable the vendor to use the new encryption key and to store a new Clearing server signature.

The "message to be authenticated" fields hold a phrase, which is stored with and without encryption. The Clearing server provides the encrypted phrase, after which either the consumer or vendor may access this phrase and attempt to decrypt it. The results are then uploaded to the Clearing server. The consumer and/or vendor will receive a confirmation for successful decryption. Depending on the encryption algorithm being used, the message may be unencrypted and the consumer or vendor would then be required to encrypt it and upload the results to the Clearing server.

Section II (Token)



The token issued by the Clearing server holds encrypted and unencrypted data in a format that permits access to specific portions of the data. The format of the token may vary from numbering of each data segment to giving each data segment a specific header. Conceptually, the token may be made to resemble a record with fields. Encryption is performed only on the Clearing server while decryption is performed by the vendor. The token contains several fields of data designed to be used for authentication. As a result the vendor using any or all of several alternative methods involving different fields may authenticate the token. For example, decrypting the Clearing server's signature field may authenticate a token.

The token is dynamically generated from the information stored in tables of the Clearing server databases, and holds encrypted and unencrypted information identifying the consumer. The fields of the token record include the following functions:

1. The consumer id field 70 is a unique key representing the consumer. It is a primary key field in the consumer record of the Clearing server database. The Clearing server creates the consumer id field 70 for each consumer when the consumer purchases the token.

2. A database alias field 72 is a reference name to identify to the Clearing server database where this consumer's record resides. Since there might be several Clearing server databases, this field lets the Clearing server know which Clearing server database to contact in order to obtain consumer information with which to create or update the token.

3. An encrypted consumer id field 74 is only readable by the Clearing server, which can decrypt this field and check it against the unencrypted consumer id to make sure the token is authentic.

4. An encrypted date and time (Greenwich Mean Time) field 76, stores information when the token was created. Every time the Clearing server modifies the token, this field 76 is updated in the token and on the Clearing server database. The token may be given a fixed life expectancy after which time it is removed from the memory of the consumer. In such cases, if for example the token cannot have a life longer than 4 hours for example, but the time field 76 shows that the token is, e.g., 12 hours old, then the token is not genuine.
5. A token amount field 78 is modified by the vendor and may be read by the consumer.
6. An encryption key code field 80 is a code for the key used by the Clearing server to encrypt the information on the token. The token that is encrypted with an old encryption key when a new encryption key is being used will be detected and recognized as a fake.
7. An encrypted token amount 82 is used to compare the value of this field with the value of the unencrypted token amount field 78. If the values don't match, it means that decrements have been made to the token of which the Clearing server is unaware. If this happens, the last vendor with whom a transaction was made is polled to obtain updates. When the transactions are obtained from the vendor the encrypted token amount in the field 82 is updated to match the unencrypted token amount in the field 78.
8. A unique ID field 84 is the id of the last vendor from whom a purchase was made. This is a unique vendor identifier maintained on the Clearing server database to identify the last vendor with whom the token was used in a transaction. The field 84 may be empty if no transaction has been made since the time that the token was created or purchased.
9. A hash value of token field 86 represents a number obtained by passing specific fields on the token through any commonly available hashing algorithms to generate a unique value. This value is encrypted. The vendor runs these specific fields through the same hashing algorithm to obtain a number to compare with the number obtained when this field 86 was decrypted. If the numbers match the token may be assumed to be authentic.
10. Clearing server's signature field 88 is used to identify the Clearing server. Decrypting this field 88 produces the Clearing server computing device's signature.
11. A unique random number field 90 holds a unique random number for each transaction session. The transaction session is the time interval starting with obtaining the token and terminating when the last purchase from a particular vendor is made. The consumer uses this number to test the key presented

by the vendor so as to determine if the vendor can be allowed to increment or decrement the value of the token, or update the entire token with a new token downloaded from the Clearing server.

12. A vendor location field 92 is used to store coordinates of the last vendor from whom a purchase was made.

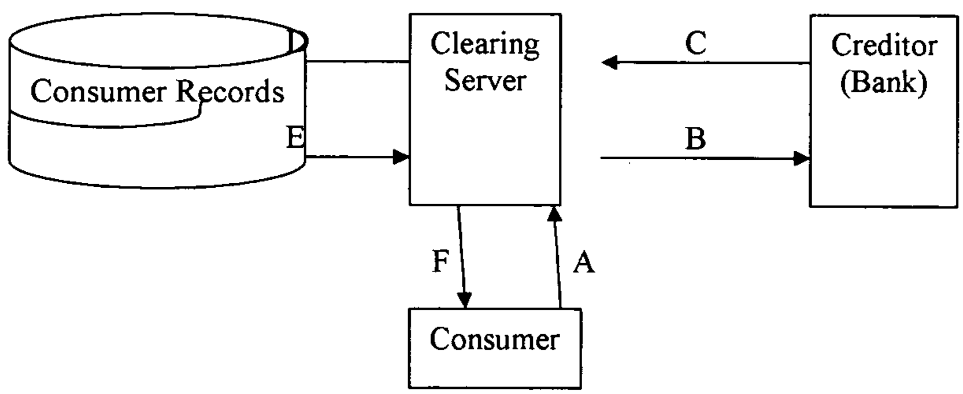
<u>The vendor modifies the token with every completed transaction, using a key obtained from the Clearing server. There are two kinds of modifications that the vendor may make. The vendor may overwrite the token with a new token copy received from the Clearing server. Alternatively, the vendor may modify the token amount by decreasing the modifiable amount field 78.</u>

Section III (Processing)

Token Creation

The consumer is permitted to select the desired token amount to be purchased. This amount can range from a fixed set of monetary units, e.g., $1, $5, $10, $20, etc., to replenishable values with upper ranges. For example, a $50 token may be issued with $100 upper range. When the value of this token is depleted it may be automatically incremented to $50 until the $100 is exhausted. Offline payments can also be used for which the prospective consumer obtains a prepaid card by mail. The information is accepted and is stored in the Clearing server database. The token is created; the customer is assigned a login name and is asked to select a password. The information is also stored in the Clearing server database.

The token creation may be achieved in the following manner.



The Clearing server receives a token-purchase request from the consumer whose payment instrument was provided to the Clearing server.

A. If the instrument is a credit card or a debit instrument, a creditor who issued that instrument is contacted with the particular details provided and the token dollar amount requested by the consumer. If the payment was made by check or money order, the Clearing of the instrument might be slower. Alternatively, if the payment was made with cash, the Clearing server will not need to contact the creditor, and will proceed to step "D." Similarly, if a prepaid card issued by the Clearing server is used, contact with the creditor is not necessary. In this situation, the Clearing server queries its own Clearing server database where information regarding the prepaid card is stored.

B. The creditor responds with a confirmation that the instrument has been debited.

C. The Clearing server will execute a "create token" query to select specific fields, comprising the token from the Clearing server database. To complete the token the Clearing server encrypts some of the fields, appends the signature and encryption key code, and appends the current time and date.

D. This token record is stored in the Clearing server database.

The token is ready to be downloaded to the consumer.

After accessing the Clearing server and submitting or modifying previously submitted information as described above, the consumer may download the token. Moreover, both the **registered** consumer and the vendor will be allowed access to download a consumer side and vendor side software to their respective computing devices.

Vendor processing

The vendor side software performs functions including the following:

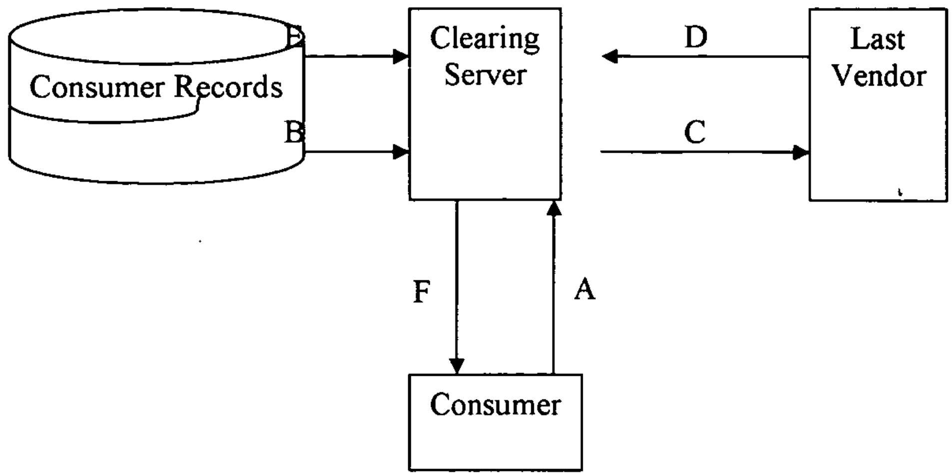
1) Communicating with the consumer side software by reading the token amount 78 and modifying the token with an encoding key obtained from the Clearing server.
2) Communicating with the Clearing server by
   a) sending it a copy of the token for further authentication;
   b) periodically uploading data on all transactions;
   c) invalidating the key received by the vendor from Clearing for modifying a consumer's token each time that customers information is uploaded to Clearing. Uploads of multiple consumer data will result in invalidation of several token keys.
   d) responding to polls and uploading aggregate purchase data on one or more consumers.
3) Maintaining the local vendor database of all transactions conducted. In this vendor database consumers are represented without revealing their personal or credit information.
4) Performing "tiered" authentication of the tokens by performing a single check or multiple checks of the token depending on the price or value of the transaction. This may involve decrypting certain fields of the token and/or running specific token data through a hash function. This hashing process is first performed on the Clearing server and the results are encrypted and written unto the token. The vendor decrypts the result and compares it with its own hash output. If the results differ the token is not genuine. The vendors who want to increase protection for their goods or services may set the vendor side software to perform multiple token authentication. This will generally produce an increase in processing time.

Consumer processing

The consumer side software performs the functions of:

1. Maintaining the token in memory and may store an encrypted token balance on disk of the consumer's computing device.
2. Displaying the remaining value or dollar amount of the token. This amount may be displayed as a counter on the Internet browsing program used by the consumer.
3. Communicating with vendor side software residing on the computing device of the vendor in negotiating payment.
4. Verifying genuineness of a vendor increment and decrement encoding "key."
5. Permitting the vendor side software to decrement or increment token using the verified encoding "key," and to overwrite the entire token.
6. Ensuring that the cash amount, incremented or decremented per transaction by the vendor, corresponds to the price displayed for the particular item or service on the computing device or website used by vendor.

Clearing server processing



The Clearing server contacts the vendor that last transacted with the consumer to determine if any transaction took place between them. The last vendor field 92 of the token identifies such vendor. The Clearing server checks the corresponding field in its Clearing server database, if the checked field is blank, the token has not yet been used to make any purchases and therefore vendors are not contacted.

A. The consumer contacts the Clearing server to obtain a previously purchased token.

B. The Clearing server checks its Clearing server databases to retrieve that consumer's token and to determine if any vendor has to be contacted to obtain token updates.

If no purchase has been made by the consumer using his/her token, since it was last updated, the Clearing server immediately proceeds to step "F" where the token is released to the consumer.

If a purchase has been made, the Clearing server checks its database to see if updates have been received from that vendor for such consumer purchases.

The updates are transfers of consumer purchase information including amounts of the transaction to the Clearing server. After sending that vendor a key for this particular consumer, the Clearing server must receive two updates from the vendor. If, when the update was taking place the consumer was making purchases, the update may exclude some purchases. A second upload from the vendor would therefore show additional transactions between the consumer and the vendor. If however the second update does not include any more purchases from this consumer, this would indicate that the consumer did not further transact with the vendor.

C. If no updates have occurred or if update conditions are not met the Clearing server polls the vendor for an update. If polling fails to generate information because perhaps the computing device of the vendor is down or disconnected from the Internet, or there is congestion on the Internet, then token is not released to the consumer computing device. A time limit is set for how long the vendor's computing device may be allowed to be dysfunctional, after which time the token is released and any credits made to that vendor may be temporarily or permanently ignored. However to avoid financial losses due to computer failures, mechanisms are available for communicating saved transaction data to the Clearing server via alternative data paths, e.g., point to point connections and modem dialup.

D. The Clearing server receives an upload of aggregate amount spent by the consumer at the vendor.

E. This information is used to update the token in the Clearing server database.

F. The token is then downloaded to the consumer.

Section IV (Operation)

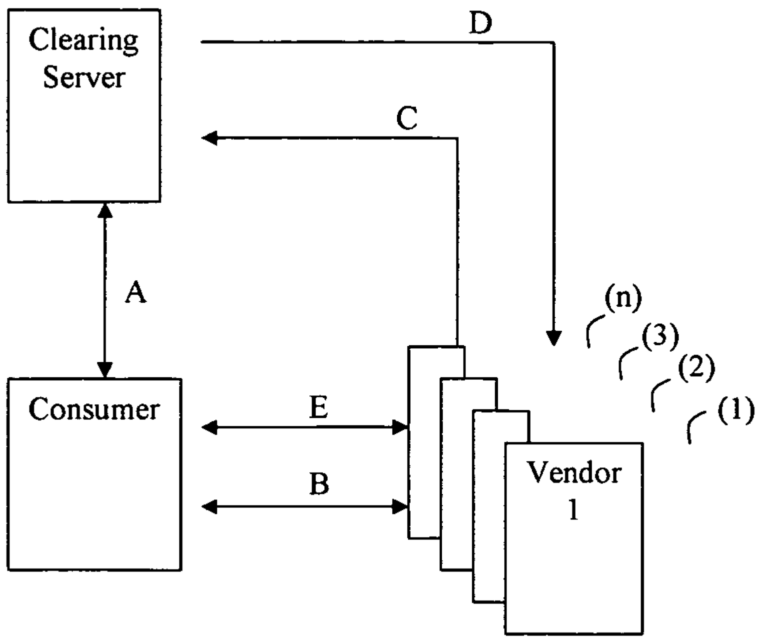
Recap

To recap the workings of this invention:

1. The consumer purchases or retrieves the token from the Clearing server. If the token has been previously used, the last vendor referenced in the token is contacted and transaction information regarding the consumer requesting the token is retrieved from the vendor database. This information verifies and assures the validity or invalidity of the token.
2. The consumer connects to a website managed or controlled by the vendor and selects goods or services to purchase and presents the token.
3. When the consumer makes the selection the vendor takes the token and compares the value amount of the token to the amount of the purchase. If the token value is insufficient the consumer is notified and the transaction is terminated.
4. If the token dollar value appears sufficient to secure the purchase, then after performing the preliminary authentication of the token, the vendor contacts the Clearing server to ascertain the validity of the token and to receive the decrement (or increment) key. Where the token has not been previously used for any purchase since it was received, only a decrement or an increment key is downloaded to the vendor. Where the token has been used with a least one other vendor then a key as well as a new token, is downloaded. The Clearing server determines that the token has been previously used with a different vendor if the encrypted (original) monetary value of the token and its actual present value do not match. The vendor then passes the information to the consumer and saves the transaction information to its database. The consumer updates its token and is ready to proceed with further purchases, continuing at step 2 or step 1 when the token value is exhausted.

Overview

Interactions between the transacting parties, the Clearing server the vendors and the consumer, from the individual consumer perspective.



A. The consumer connects to the Clearing server, purchases the token of a particular value, by paying with a credit, debit, or prepaid card. Alternatively the payment may be made with cash, check, and/or money order. First time users would be requested to enter specific consumer information such as first name, last name, email address, city, country, login name. Consumers who pay with a previously purchased prepaid card would enter a number on the card. This card number represents an anonymous consumer record pre-created on the Clearing server database. When paying with the pre-paid card, the consumer will not be asked to enter personal information. The prepaid card may reference an email address, which the consumer can access using the prepaid card number. This email address may be used to send and receive information from the Clearing server as well as for transaction dispute resolution.

All the consumer's information is saved in the Clearing server databases. The Clearing server creates a token specific to the consumer by selecting information from that consumer's information. The information on the token comprises encrypted consumer information, token amount, authentication data such as a random number, a date/time field, and the Clearing server's encrypted signature.

The token and the consumer side software, i.e., for first-time users, is downloaded to the consumer. The consumer side software loads the token into memory, making it inaccessible to other applications executing on the computing device used by the consumer. The token balance amount is displayed to the consumer.

B. The consumer, using commonly available Internet browsing programs, visits any participating vendor. The participating vendor must have registered with the client server and installed the vendor side software. The consumer may enter into commercial transactions, e.g., for purchases of goods and services, with the vendor. Transactions may be performed for any range of items. Some items may be obtained immediately, e.g., downloading of digital multimedia files and spending time in a chat room, other items may be delivered in the future, e.g., books, or services consumed over time.

The transaction prices may be displayed by the vendors next to each product. A description, small version, or a thumbnail of the product may be provided with its price. The consumer may negotiate the price with the vendor or accept it by clicking on the description or the thumbnail. The vendor side software communicates with the consumer side software to retrieve the token and to perform basic authentication to determine, for example, if the token has an adequate balance for the intended transaction. The authentication is performed by checking the Clearing server's encrypted signature 88. Additionally, the vendor determines if the token has been updated by the Clearing server.

C. When the consumer elects to make a purchase, the payment is made immediately, in real-time, and the token amount visible to the consumer is decremented by the amount of the purchase. The amount may be incremented if credit is provided to the consumer. The consumer side software allows the consumer to generate a record of the amount spent and the URL of the vendor where the amount was spent. After transacting with one vendor (1) the consumer may connect to other vendors (2) and repeat the actions described in step "B."

A copy of the token is forwarded to the Clearing server together with a request for a decrement key. Where the vendor wishes to credit the token, a copy of the token is forwarded with a request for an increment key. The crediting transaction requires that the vendor maintain a credit balance in its account on the Clearing server. This credit may be created identical to the debiting of the token.

The Clearing server authenticates the token by decrypting and checking the encrypted fields, including the date/time field 76, and by comparing data on the token with matching data in the Clearing server database. If the token is determined to be authentic, the Clearing server writes the vendor ID 84 of the requesting vendor on the token and unto the Clearing server database. A key with which the vendor can decrement or increment the token is then created. This key may be

created using any commonly available random number generator. This key may allow the vendor to decrement or increment the token amount for contiguous transactions with the same consumer without re-authentication.

Contiguous transactions may include several purchases performed one after another from the same vendor. The consumer may not (a) shut down their computing device between purchases, or (b) make intervening transactions with other vendors. Any such action breaks the continuity and a new key will be required.

(a) Where the consumer (1) shuts down the computing device, the token must be re-obtained from the Clearing server. At retrieval the token is assigned a new random number.

(b) If the consumer makes a purchase from a different vendor (2), that vendor (2) downloads a key from the Clearing server, which first overwrites the existing token with a new token and then decrements the new token. The new token, which is downloaded along with the key, will have a new random number. In other words, the second vendor (2) first overwrites the consumer's existing token before decrementing it.

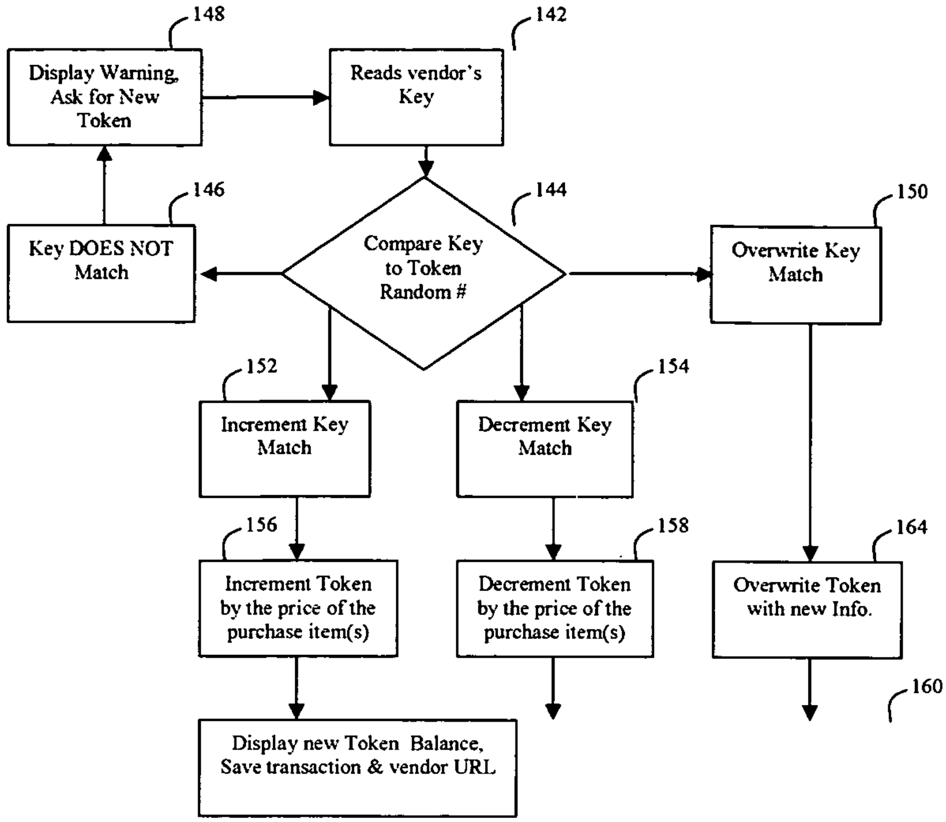
D. The vendor receives the decrement key and notifies the consumer side software to decrement the token value. The consumer side software uses an embedded algorithm to compare the decrement key with the random number 90. If the entire key matches, the entire token is overwritten. If only specific positions on the key match then only the token amount is decremented. The following example illustrates the conceptual basis of key algorithms a variety of which can be used in conjunction with the present invention. The consumer side software checks the key and determines whether to permit an overwrite operation, a decrement operation, or an increment operation.

If the token has a random number "123456789" then the overwrite key could be a matching number "123456789," while the decrement key could be "123z56789" where the fourth number "4" is non-matching and is replaced with "z." An increment key could be "1234567q9," where the eighth number "8" is non-matching and is replaced with "q".

E. After token is decremented, the consumer is allowed to receive the goods, e.g., download a file. Just before the token value is decremented, the consumer may be prompted to enter the password, if the

correct password is entered the token is decremented and the goods are released. This feature may be turned off or on, as the consumer requires. For each transaction, the consumer side software only permits decrements or increments of the token in the amount of the price received from the vendor. This price is either displayed or negotiated with the vendor. Following the transaction, the new token balance is displayed to the consumer.

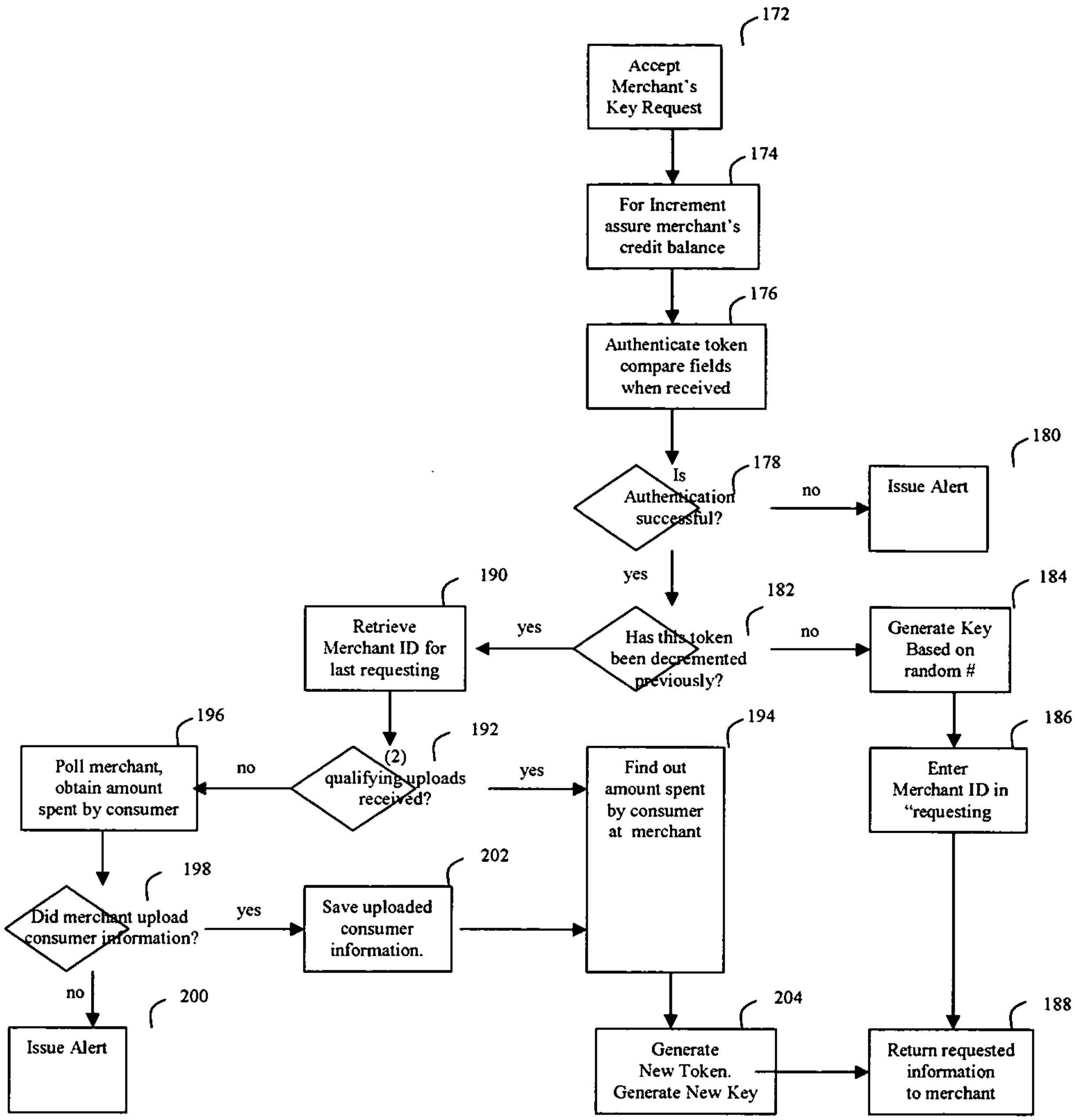
Consumer Side Software



Vendor provides a key, which, if correct, enables the consumer side software to perform updates of the token. In step 142 the key is accepted and is matched against the random number provided on the token. The way the key matches the random number determines whether an overwrite, a decrement or an increment operation will be performed. In step 144 the key is matched, and the type of operation is determined. Step 146 indicates that the key does not match at all. In step 148 a warning is displayed or the token is locked after several wrong keys are presented by the vendor. The URL of the vendor is saved and a warning message is displayed. The locking of the token halts all operations for a time if there are repeated failed attempts to increment, decrement, or overwrite the token. The duration of such lockout is adjustable.

An overwrite key will be matched in step 150 and the token will be overwritten in step 164 when the vendor will provide an overwrite key and a new token. This situation will arise when the token has been previously used with a different vendor and the original face value of the token and its actual present value do not match. Otherwise, the increment/decrement keys will be matched in steps 152, 154 and the token will be managed accordingly in steps 156, 158. In step 160 the new token balance will be displayed and the visited vendor URLs or IDs and purchases made will be saved. The consumer may be required to enter password for each transaction or to enter a password for transactions exceeding a certain amount.

Providing the Key to Enable the Performance of the Token Updates

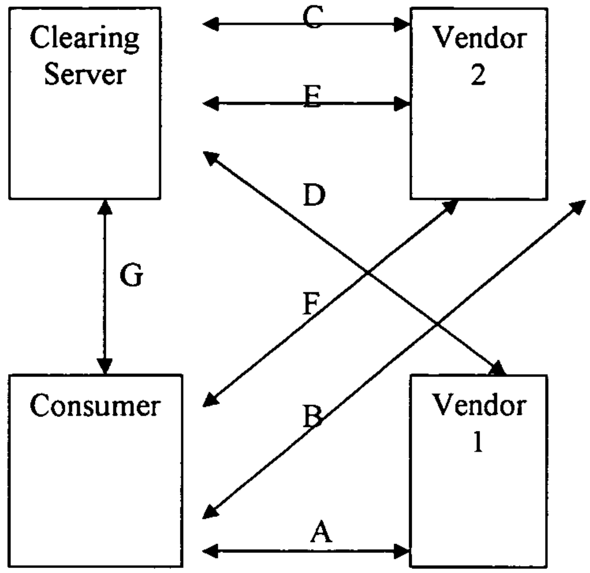


In step 172, the Clearing server receives the request for the key from the vendor. If the request is for the increment key, to credit the consumer, in step 174, vendor's balance with the Clearing server is checked to verify sufficiency of funds. In step 176, the token provided by the vendor along with the request for the key is authenticated. The authentication may be achieved by comparing all or select fields of the vendor provided token and the original fields of the Clearing server database records used to create the token provided to the consumer. If in step 178, it is determined that the authentication is not successful, appropriate parties will be alerted to that fact in step 180.

If the token is authentic, in step 182 it is determined if it has been previously decremented. This is achieved by comparing the present value of the token and the encrypted original value, another way to achieve this is to dedicate a bit on the token, which is set on or off when the token is used. If the token has not been previously decremented, the requested increment/decrement key is generated in step 184, the vendor information is stored in step 186, and in step 188, the key is given to the vendor.

If the token has been previously decremented, the last requesting vendor ID is retrieved in step 190, and the Clearing server database is checked to see if there were sufficient qualifying uploads from that vendor. If there were, the amount spent by the consumer is ascertained in step 194, and in step 204 a new token as well as the requested key is generated and returned to the vendor in step 188. Please recall that the vendor will in its turn forward this information to the consumer. The new token will be passed along to the consumer with an overwrite key. If there were not sufficient qualifying uploads, the vendor is polled in step 196 to obtain amount spent by the consumer. If the vendor did not upload consumer information, as determined in step 198, appropriate parties will be alerted to that fact in step 200. Otherwise the consumer information is stored in step 202 and the processing of program 170 continues from step 194.

Consumer Loses the Token Before Completing Transactions



A. The consumer leaves vendor (1) after making purchases and connects to another vendor (2). The field holding the modifiable token balance 78 has been decreased by the amount of the purchase from the last vendor (1). All purchases are also recorded by the vendor side software of the last vendor (1) in its database and on the token via decrements performed with the decrement key. All credits are likewise recorded in the vendor database and applied to the token using the increment key. At this point the Clearing server has not been provided with any information and is unaware of how much the consumer has spent. However, the token amount field 78 modified by the last vendor (1) and the encrypted amount field 82 are different.

B. The consumer then connects to another vendor (2) to make additional transactions. The vendor side software authenticates the token.

C. The Clearing server is contacted by the vendor (2), and provided with a copy of the token and a request for a decrement key with which to decrement the token value. Alternatively, as mentioned above an increment key may be requested. The Clearing server authenticates the token and then decrypts the "encrypted token amount field" 82. If this amount is different from the modifiable token amount field 78 the decrements have been performed.

D. Using the ID of the last vendor field 84 value, in the present example the first vendor (1), the Clearing server polls vendor (1) and obtains total amount of purchases made by the consumer. This amount is then subtracted from the original encrypted amount field 82. The new amount should match the amount in the modifiable amount field 78. This amount is encrypted and is written back to the encrypted token amount field 82. The Clearing server generates a new token with a new correct token amount, 78 a new time stamp 76, having ID of vendor (2) in the last vendor field 84, and a new

random number 90. A decrement key is then generated using this random number and the last vendor ID, the consumer and vendor databases are updated with the new data. Where a comparison of encrypted and modifiable amount fields 78, 82 shows that an increment has occurred, a similar process is pursued except that additionally, previous vendor's (1) account is decreased by the amount of the increment or credit applied to the consumer.

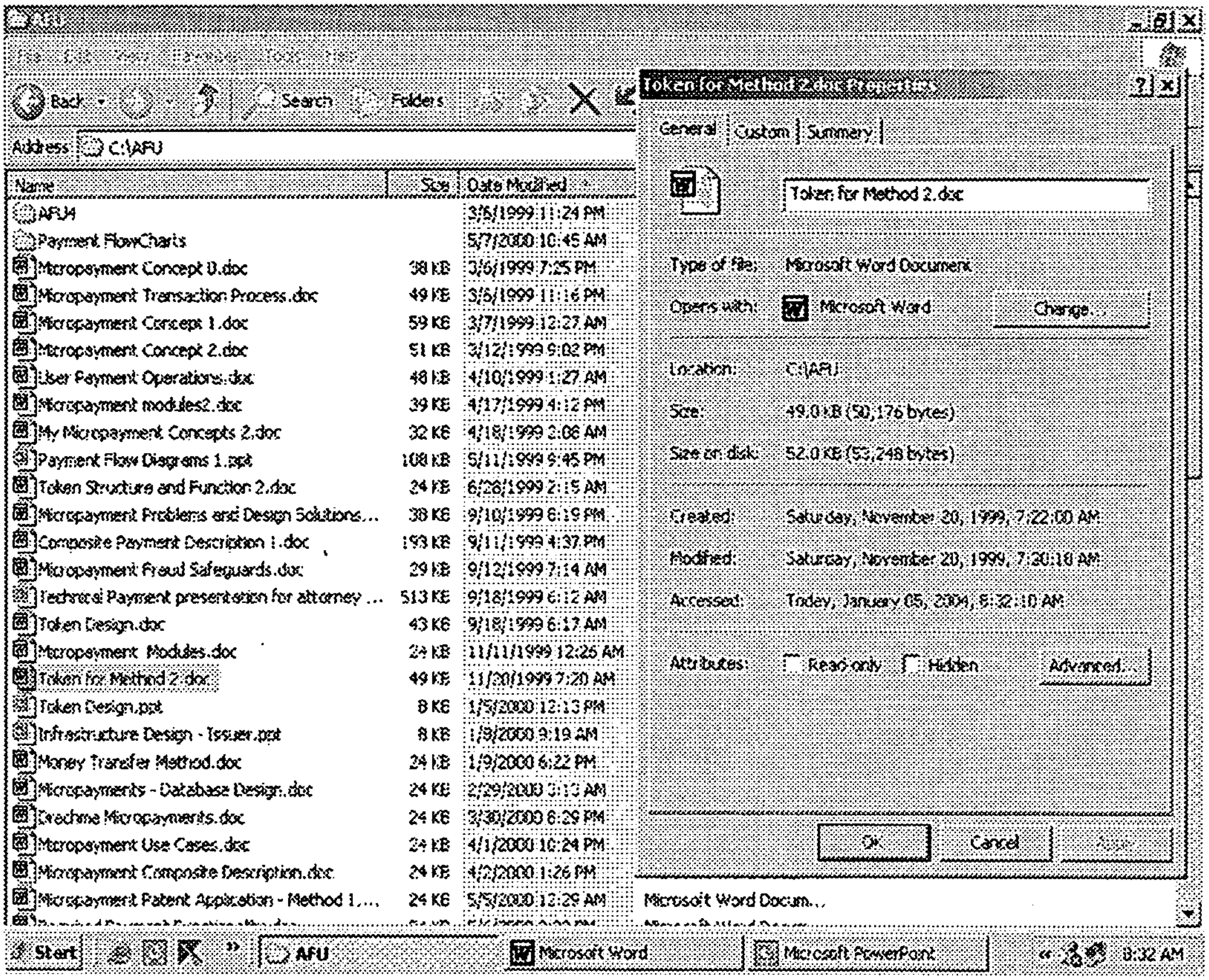
The vendors are required to periodically upload all their transactional data so that the Clearing server database may be accurately updated and new tokens created on the Clearing server. If this upload was performed immediately prior to the inquiry described in step "D" it may be unnecessary to poll the last vendor (1).

At least two uploads must have been received from the previous vendor (1) since the creation of the token, i.e., uploads after the time stamped in field 76. The first upload must have an entry for the consumer and the second must not. That is because, if the consumer was still making purchases from vendor (1) after the first upload, then that first upload would not capture all purchases made from vendor (1). However if a second upload, has no entries from the consumer, but the first upload does, it means that the consumer made no purchases after the first upload. Therefore, the data from the first upload is reliable and may be used to create the token and update the Clearing server database.

E. The token and the decrement key are then downloaded to the vendor (2).

F. The vendor side software then contacts the consumer side software and "shows" the new key. The token on the consumer is overwritten and then the new token amount field 78 is decremented by the amount of the purchase. Additional decrements may then be performed without contacting the Clearing server for additional consequential purchases from the last vendor (2).

G. For various reasons, the consumer may wish to begin a new session, e.g., a shutdown of a computing device has occurred. The consumer must connect to the Clearing servers and retrieve their token. The Clearing servers check their Clearing server databases to identify the last vendor visited. If sufficient uploads have been received from that vendor, please see discussion of sufficient uploads above, the Clearing server may use the information from those uploads to create a new token for the consumer. If the uploads were not sufficient, the Clearing server polls the last vendor and uploads total amount spent by the particular consumer. In either case the information is used to update the token amount 78 and the Clearing server database. Thereafter a new token is created.


AFU
Back
Search
Folders
Address C:\AFU
Name
Size
Date Modified
AFU4 3/5/1999 11:24 PM
Payment FlowCharts 5/7/2000 10:45 AM
Micropayment Concept 0.doc 38 KB 3/6/1999 7:25 PM
Micropayment Transaction Process.doc 49 KB 3/5/1999 11:16 PM
Micropayment Concept 1.doc 59 KB 3/7/1999 12:27 AM
Micropayment Concept 2.doc 51 KB 3/12/1999 9:02 PM
User Payment Operations.doc 48 KB 4/10/1999 1:27 AM
Micropayment modules2.doc 39 KB 4/17/1999 4:12 PM
My Micropayment Concepts 2.doc 32 KB 4/18/1999 2:08 AM
Payment Flow Diagrams 1.ppt 108 KB 5/11/1999 9:45 PM
Token Structure and Function 2.doc 24 KB 6/28/1999 2:15 AM
Micropayment Problems and Design Solutions… 38 KB 9/10/1999 6:19 PM
Composite Payment Description 1.doc 193 KB 9/11/1999 4:37 PM
Micropayment Fraud Safeguards.doc 29 KB 9/12/1999 7:14 AM
Technical Payment presentation for attorney … 513 KB 9/18/1999 6:12 AM
Token Design.doc 43 KB 9/18/1999 6:17 AM
Micropayment Modules.doc 24 KB 11/11/1999 12:26 AM
Token for Method 2.doc 49 KB 11/20/1999 7:20 AM
Token Design.ppt 8 KB 1/5/2000 12:13 PM
Infrastructure Design - Issuer.ppt 8 KB 1/9/2000 9:19 AM
Money Transfer Method.doc 24 KB 1/9/2000 6:22 PM
Micropayments - Database Design.doc 24 KB 2/29/2000 3:13 AM
Drachma Micropayments.doc 24 KB 3/30/2000 6:29 PM
Micropayment Use Cases.doc 24 KB 4/1/2000 10:24 PM
Micropayment Composite Description.doc 24 KB 4/2/2000 1:26 PM
Micropayment Patent Application - Method 1.… 24 KB 5/5/2000 12:29 AM
Token for Method 2.doc Properties
General
Custom
Summary
Token for Method 2.doc
Type of file: Microsoft Word Document
Opens with: Microsoft Word
Change...
Location: C:\AFU
Size: 49.0 KB (50,176 bytes)
Size on disk: 52.0 KB (53,248 bytes)
Created: Saturday, November 20, 1999, 7:22:00 AM
Modified: Saturday, November 20, 1999, 7:20:18 AM
Accessed: Today, January 05, 2004, 8:32:10 AM
Attributes: Read-only Hidden
Advanced...
OK
Cancel
Microsoft Word Docum...
Start
AFU
Microsoft Word
Microsoft PowerPoint
8:32 AM

Token

The token issued by the clearing server 12, holds encrypted and unencrypted data in a format that permits access to specific portions of the data. The format of the token may vary from sequential binary data, to numbering of each data segment, to giving each data segment a specific header. Conceptually, the token may be seen as a record with fields. Encryption of token fields is performed only on the clearing server 12 while decryption is performed by the merchant 14. Any robust commercially available encryption algorithm - symmetric or asymmetric - may be used to encrypt and decrypt token data although symmetric encryption is computationally cheaper. The token contains several fields of data designed to be used for authentication by either the clearing server 12 or the merchant 14 using any or all of several alternative methods involving different fields on the token 68. For example, decrypting the clearing server's signature field may authenticate a token to the merchant 14.

Each token 68 may hold encrypted information, which may include at a minimum, the token ID, the token's cash value amount, date and time, and a unique identifier. The token resides in memory 34 of the computing devices 14, and 16 of the merchant 14 and is therefore difficult to access or tamper with. Moreover, on moving from one merchant to the next, the token 68 is essentially destroyed and a new token 68 created. Encryption and decryption of the token is performed by software, is invisible to the consumer and merchant 14, and does not depend on any particular encryption algorithm although secure encryption methods which use few processing cycles will allow both the clearing server 12 and the merchant 14 to process more transactions.

The token 68 is dynamically generated from the information stored in tables of the clearing server databases 13, and holds encrypted and unencrypted information identifying it as created by the clearing server 12. A symbolic representation of a token is provided in Figure 1 below. The fields of the token record 68 address the following functions:

68

70	72	74	76	78	80	82	84	86	88	90	92
Consumer ID	Database Alias	Encrypted Consumer ID	Encrypted Time and Date (GMT)	Token Amount	Encryption Key Code	Encrypted Token Amount	Merchant ID (Merchant of Last Purchase)	Encrypted Hash value of Token	Encrypted Clearing Server Signature	Unique Random Number	Last Merchant URL

Figure 1

1. The token ID field 70 is a unique key representing the consumer. It is a primary key field in the consumer record of the clearing server database 13 (Figure 2). The clearing server 12 (Figure 2 below) creates the token ID field 70 for each consumer 16 when the consumer purchases the token 68.

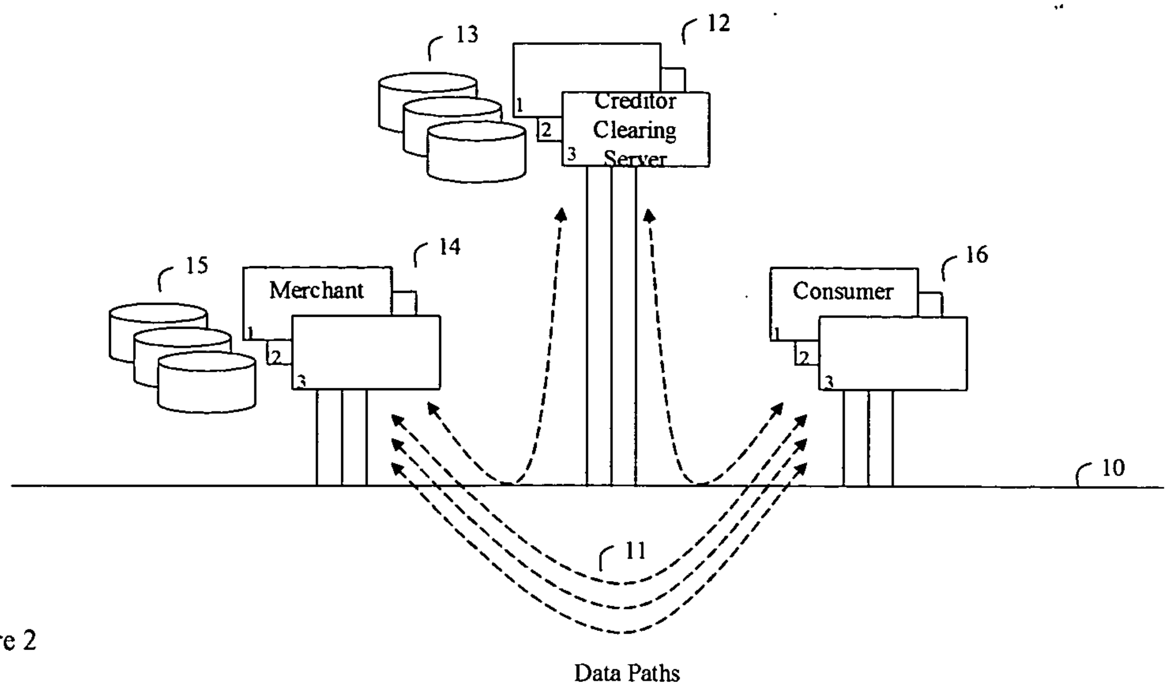


Figure 2

2. A database alias field 72 is a reference name that identifies to the merchant the clearing server database 13 in which this token's transaction records reside. Since there might be several clearing server databases 13, this field is used to direct the merchant 14 to upload transaction information generated on the token 68 to a different clearing server 12.

3. An encrypted token ID field 74 is only readable by the clearing server 12, which can decrypt this field and check it against the unencrypted consumer id to ensure the token 68 is authentic. Since the merchant 14 includes the token in the database records it uploads to the clearing server 12, the clearing server 12 can authenticate the token 68 using this or other fields to confirm that the database records received is authentic.

4. Date and time (Greenwich Mean Time) field 76, stores information on when the token 68 was created. Every time the clearing server 12 creates a token 68, this field 76 is updated in the token 68 and on the clearing server database 13. The token 68 may be given a fixed life expectancy after which time it is removed from the memory 34 of the merchants machine and further transactions halted. It may also be used for authentication. If the token 68 cannot have a life longer than 4 hours for example, but the time field 76 shows that the token 68 is, e.g., 12 hours old, then the token 68 is not genuine.

5. A token amount field 78 is read by the merchant 14 (Figure 2) and used as the beginning balance in its local database 12 on which transactions on the token 68 are recorded.

6. An encryption key code field 80 is a code for the key used by the clearing server 12 to encrypt the information on the token 68. The token 68 that is encrypted with an old encryption key when a new encryption key is being used will be detected and recognized as a fake.

7. An encrypted token amount 82 is used by the merchant 14 for authentication. This field is compared with the value of the unencrypted token amount field 78. If the values do not match, it means the token is not genuine.

8. A merchant ID field 84 is the unique ID of the merchant 14 who leased the token from the clearing server 12. This unique merchant identifier maintained on the clearing server database 13 enables the clearing server 12 to confirm that a token 68 received from a merchant 14 was actually sent to that merchant 14.

9. A hash value of token field 86 represents a number obtained by running some or all of the fields on the token 68 through any commonly available hashing algorithm that generates a unique value. This value is encrypted. The merchant 14 runs the same fields through the same hashing algorithm to obtain a number which is compared with the number obtained when this field 86 was decrypted. If the numbers match the token 68 may be assumed to be authentic. Ideal fields to hash would be amount, Token ID and Time.

10. Clearing server's signature field 88 is used to identify the clearing server 12. Decrypting this field 88 produces the clearing server computing device's signature.

11. A unique random number field 90 holds a unique random number for each transaction session. The transaction session is the time interval starting with obtaining the token 68 and terminating when the last purchase from a particular merchant is made.

12. A merchant database hash key field 92 holds a hash key, used by the merchant 14, to generate a unique code for each new database record, which is then stored in a field in the preceding record. Every record therefore has a unique code that "points" to the record after it in the database index. Insertion or deletion of a record requires re-computation and update of the unique code field, a process that cannot be performed without the hash key. Any unauthorized insertions or deletions into the merchant database 15 can be readily identified because the unique code field above the inserted or deleted record will be incorrect. Merchant database hash protection is described in later. The token 68 has several security features, all of which do not have to be used at the same time. However, the present invention may be arranged such that the higher the dollar amount of the purchase, the more checks are performed on the token 68.

The merchant 14 attaches the token 68 with every database transactions upload to the clearing server 12, and the clearing server 12 authenticates the token before accepting the tables provided by the merchant 14. (We don't have to use all the authentication mechanisms outlined here. I think we simply need fields 70 –Token ID, 76-Date and Time, 78-Token amount, 82-Encrypted Token amount, 92-Merchant hash key as the default for authentication).

The merchant 14 uses merchant side software to perform the transactions on its computer. The merchant may be allowed to download merchant side software to his computing devices from the clearing server. Alternatively, the merchant side programs may be provided on any computer readable medium, e.g., diskettes.

The merchant side software performs functions including the following:

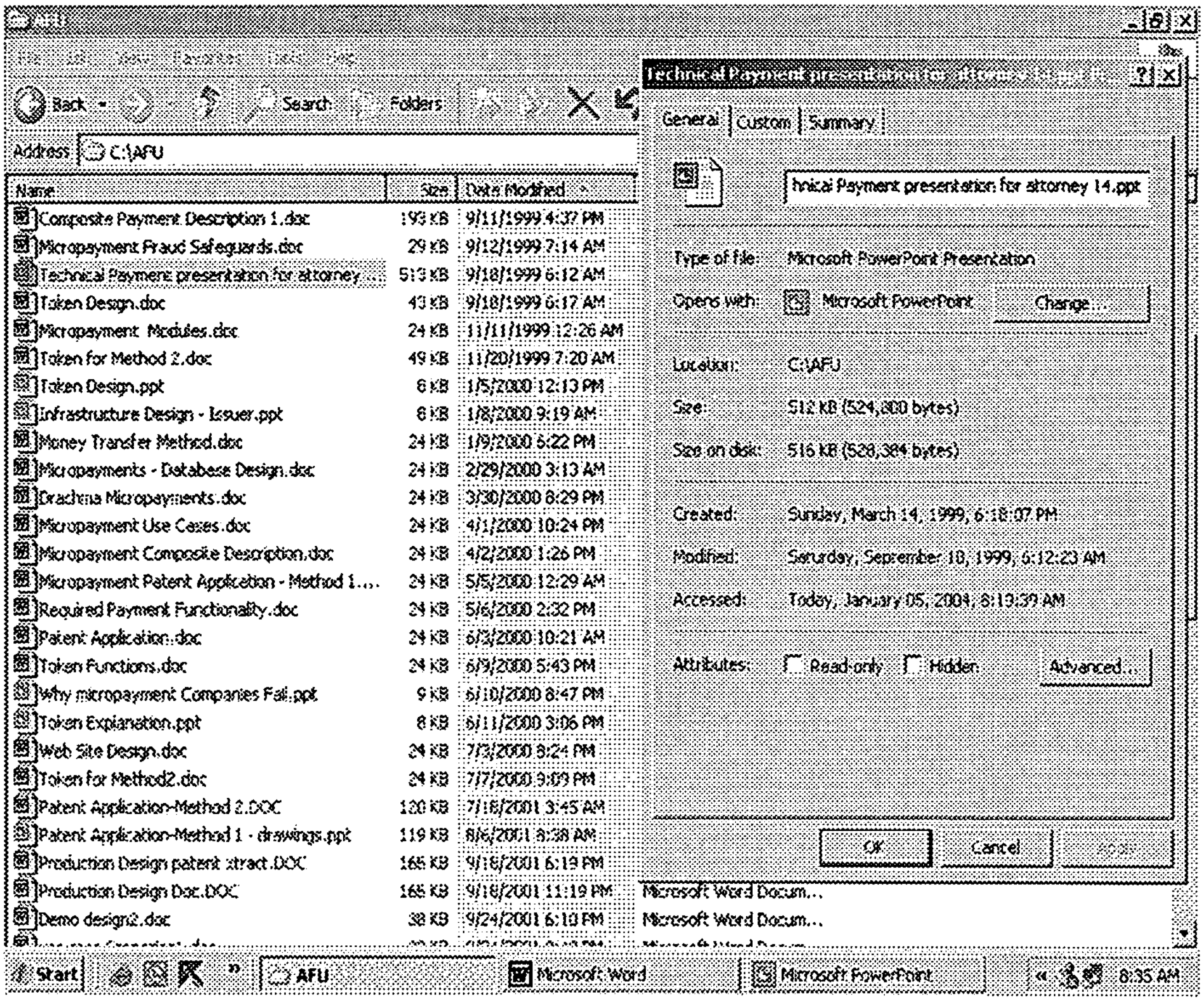
1) Communicating with the server side software, downloading the token 68, and reading the token amount 78 and with a decryption key obtained from the clearing server 12. This allows the clearing server 12 to make a token 68 readable only by the requesting merchant 14.

2) Communicating with the clearing server 12 by

    a) sending it a copy of the token 68 for further authentication if necessary;
    b) periodically uploading data on all transactions to the clearing server; and
    c) responding to polls and uploading aggregate purchase data on one or more tokens.

3) Maintaining the local merchant database 15 (Figure 2) of all transactions conducted. In this merchant database 15, consumers are represented as token IDs which avoids revealing their personal or credit information.

4) Performing "tiered" authentication of the tokens 68 by performing a single check or multiple checks of the token 68 depending on the token amount or value of the transaction. This may involve decrypting encrypted fields of the token 68 and comparing them against unencrypted fields, and/or running specific token data through a hash function. This hashing process is first performed on the clearing server 12 and the results are encrypted and written unto the token 68. The merchant 14 decrypts the result and compares it with its own hash output. If the results differ the token 68 is not genuine. The merchants who want to increase protection for their goods or services may set the merchant side software to perform multiple token authentication. This will generally produce an increase in processing time.


Back
Search
Folders
Address C:\AFU
Name
Size
Date Modified
Composite Payment Description 1.doc 193 KB 9/11/1999 4:37 PM
Micropayment Fraud Safeguards.doc 29 KB 9/12/1999 7:14 AM
Technical Payment presentation for attorney ... 513 KB 9/18/1999 6:12 AM
Token Design.doc 43 KB 9/18/1999 6:17 AM
Micropayment Modules.doc 24 KB 11/11/1999 12:26 AM
Token for Method 2.doc 49 KB 11/20/1999 7:20 AM
Token Design.ppt 6 KB 1/5/2000 12:13 PM
Infrastructure Design - Issuer.ppt 8 KB 1/8/2000 9:19 AM
Money Transfer Method.doc 24 KB 1/9/2000 6:22 PM
Micropayments - Database Design.doc 24 KB 2/29/2000 3:13 AM
Drachma Micropayments.doc 24 KB 3/30/2000 8:29 PM
Micropayment Use Cases.doc 24 KB 4/1/2000 10:24 PM
Micropayment Composite Description.doc 24 KB 4/2/2000 1:26 PM
Micropayment Patent Application - Method 1.... 24 KB 5/5/2000 12:29 AM
Required Payment Functionality.doc 24 KB 5/6/2000 2:32 PM
Patent Application.doc 24 KB 6/3/2000 10:21 AM
Token Functions.doc 24 KB 6/9/2000 5:43 PM
Why micropayment Companies Fail.ppt 9 KB 6/10/2000 8:47 PM
Token Explanation.ppt 8 KB 6/11/2000 3:06 PM
Web Site Design.doc 24 KB 7/3/2000 8:24 PM
Token for Method2.doc 24 KB 7/7/2000 9:09 PM
Patent Application-Method 2.DOC 120 KB 7/16/2001 3:45 AM
Patent Application-Method 1 - drawings.ppt 119 KB 8/6/2001 8:38 AM
Production Design patent xtract.DOC 165 KB 9/16/2001 6:19 PM
Production Design Doc.DOC 165 KB 9/18/2001 11:19 PM
Demo design2.doc 38 KB 9/24/2001 6:10 PM
Microsoft Word Docum...
Microsoft Word Docum...
General
Custom
Summary
hnical Payment presentation for attorney 14.ppt
Type of file: Microsoft PowerPoint Presentation
Opens with: Microsoft PowerPoint
Change...
Location: C:\AFU
Size: 512 KB (524,800 bytes)
Size on disk: 516 KB (528,384 bytes)
Created: Sunday, March 14, 1999, 6:18:07 PM
Modified: Saturday, September 18, 1999, 6:12:23 AM
Accessed: Today, January 05, 2004, 8:19:39 AM
Attributes: Read-only Hidden
Advanced...
OK
Cancel
Start
AFU
Microsoft Word
Microsoft PowerPoint
8:35 AM

Digital Payments Invention

Technical Overview

Novel Payment Features

1. Novel digital payment Invention ideal for making micropayments.

2. Encrypted cash, called a Token, is download into a computing device and used to make payments.

3. Customers can refill the Token and reuse it as needed

4. Customers can use the Token to make money transfers.

5. The Token is Volatile: It resides in RAM

How Micropayments Work

Micropayments are based on 2 primary systems:

Notational Payment Systems

- Resembles purchase by check.

Token or Value Payment Systems

- Token is digitally encrypted cash which can be debited directly for purchases.

Problems with Notational Systems

- Not suitable for small transactions because of huge overhead costs.

- Expensive to transact in real time.

- Vendor knows consumer's identity and financial information.

Token Payment Systems

Token Systems represent cash as an encrypted digital message called a Token. The Token itself is debited for each purchase , which is cost efficient.

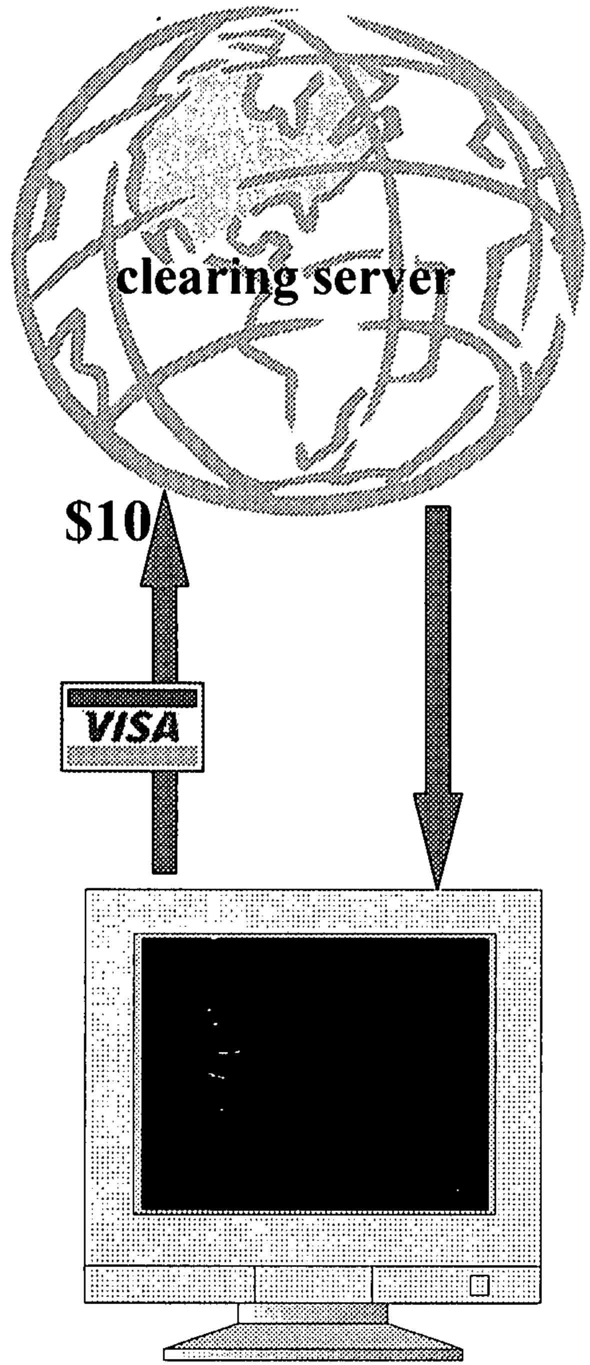
- The consumer first provides funds equal in value to the Token.
- The issuing party then creates a Token and downloads it to the consumer.
- The consumer can now make purchases with this Token which is debited directly by the vendor for every purchase

Problems with Token Systems

- **Digital Copying**: Multiple copies of one Token can be made and spent.
- **Hardware Based**: Cumbersome extra devices, e.g., smartcards, often needed by both consumer and vendor.
- **Not Scalable**: Usable ONLY for micropayments.
- **Not Refundable**: Do not allow credits or refunds.
- **Not Fungible**: Making change for a Token is problematic.

How It Works

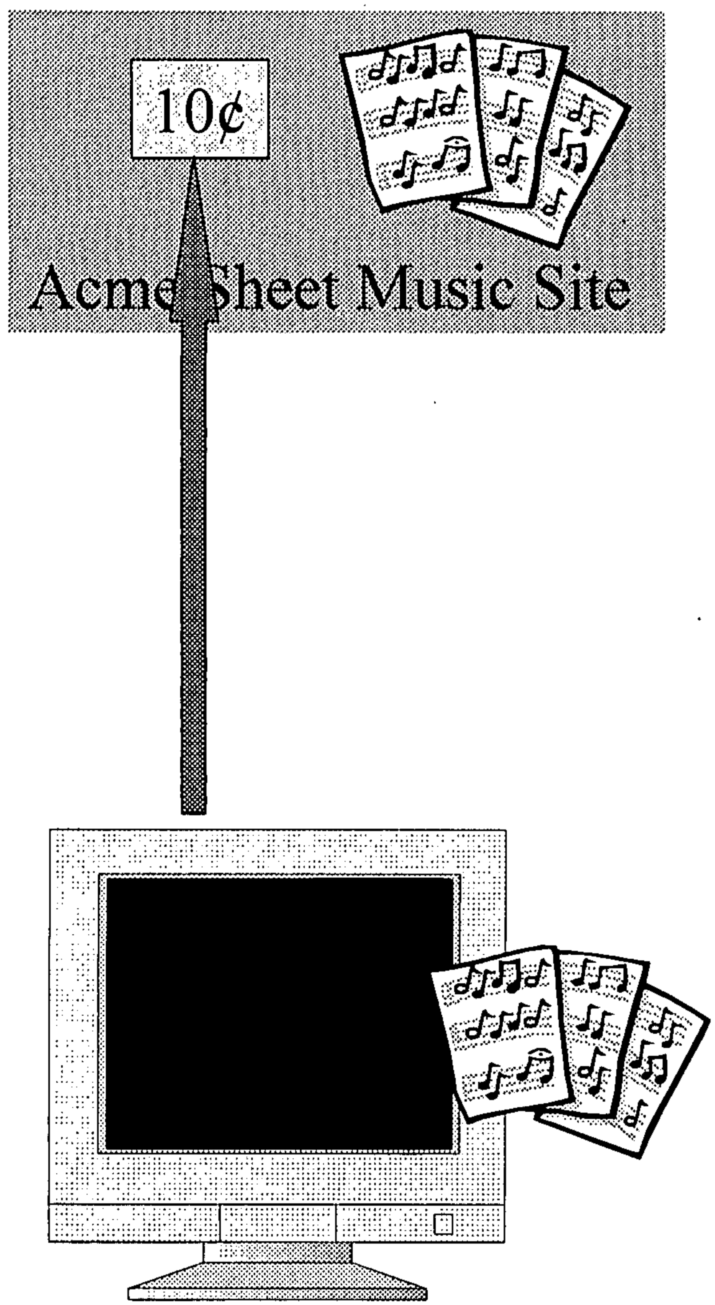
- This payment invention is a **hybrid** between Notational and Token payment systems.
- Unlike pure Token systems, the clearing server **is contacted only for the consumer's first transaction** with a merchant.
- The **merchant receives a key with which to directly debit the buyer's Token**.
- Subsequent purchases from that merchant require **no further contact with the clearing server**.
- **Tokens have a life of 2** hours and reside in memory on the consumer's computer.



Buying a Token

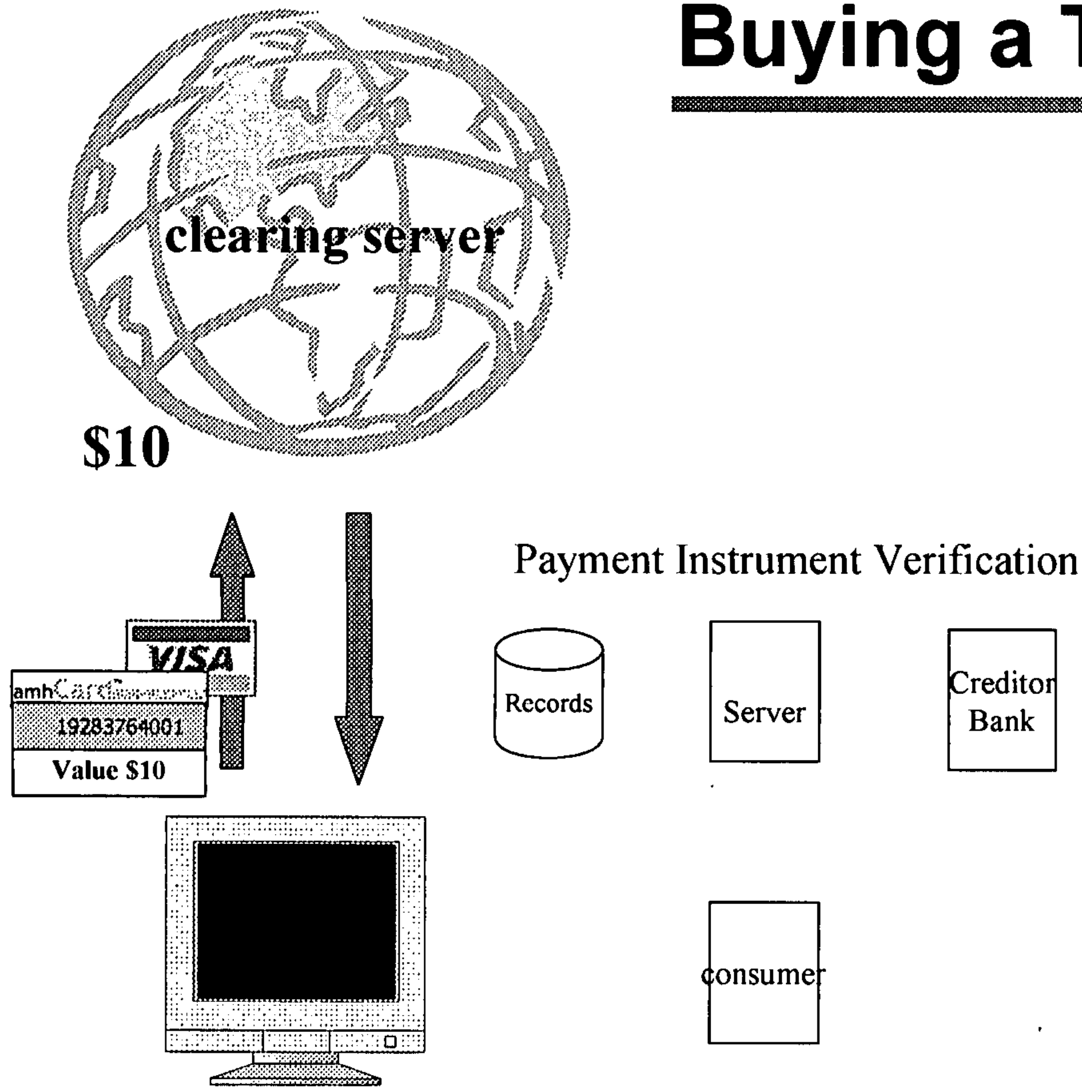
- Consumer visits clearing server Website and purchases a $10 token using a credit, debit, or prepaid card.
- $10 token appears in corner of consumer's screen.

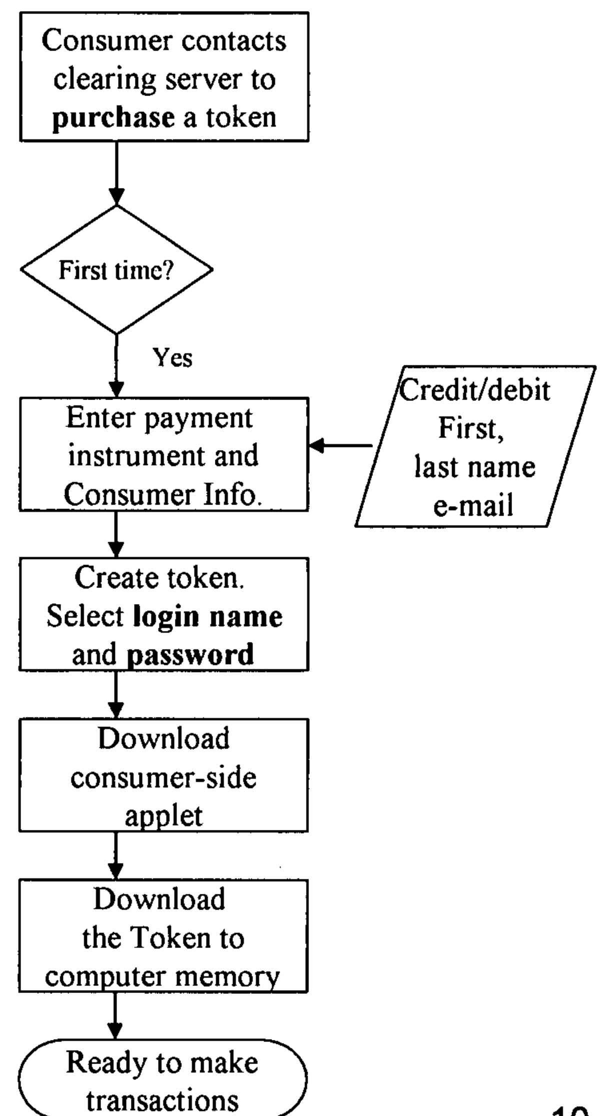
Making a purchase



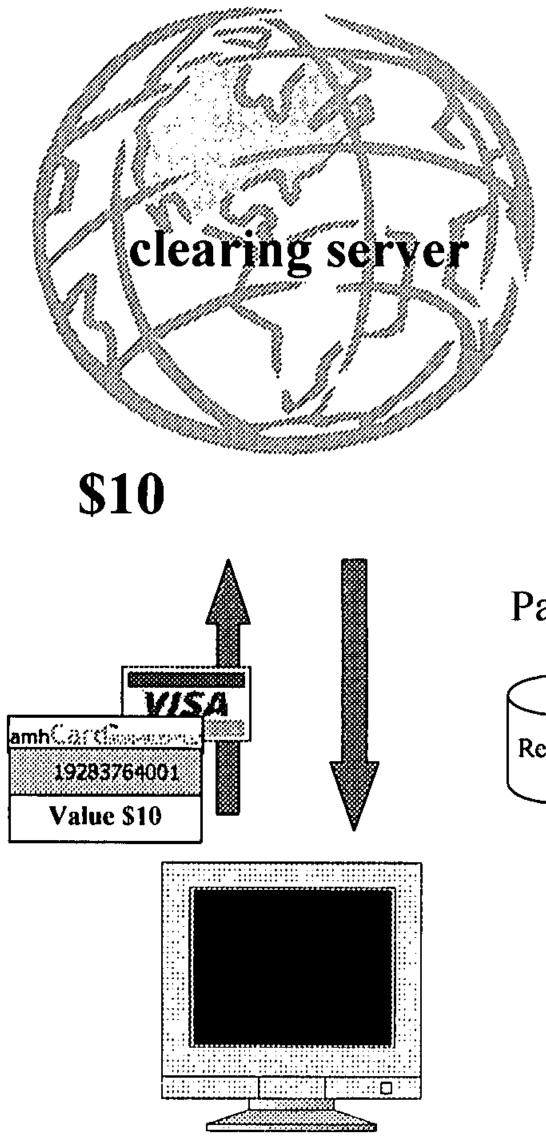
- Consumer visits Sheet Music Vendor Website and purchases sheet music by clicking 10¢ icon.
- Vendor debits token by 10¢ leaving $9.90
- Consumer downloads sheet music.

Buying a Token: Technical

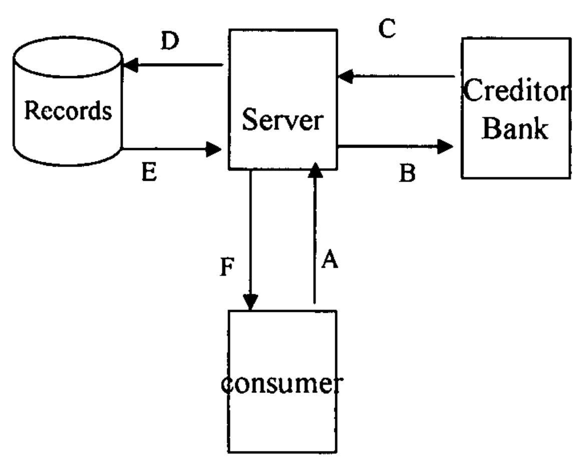


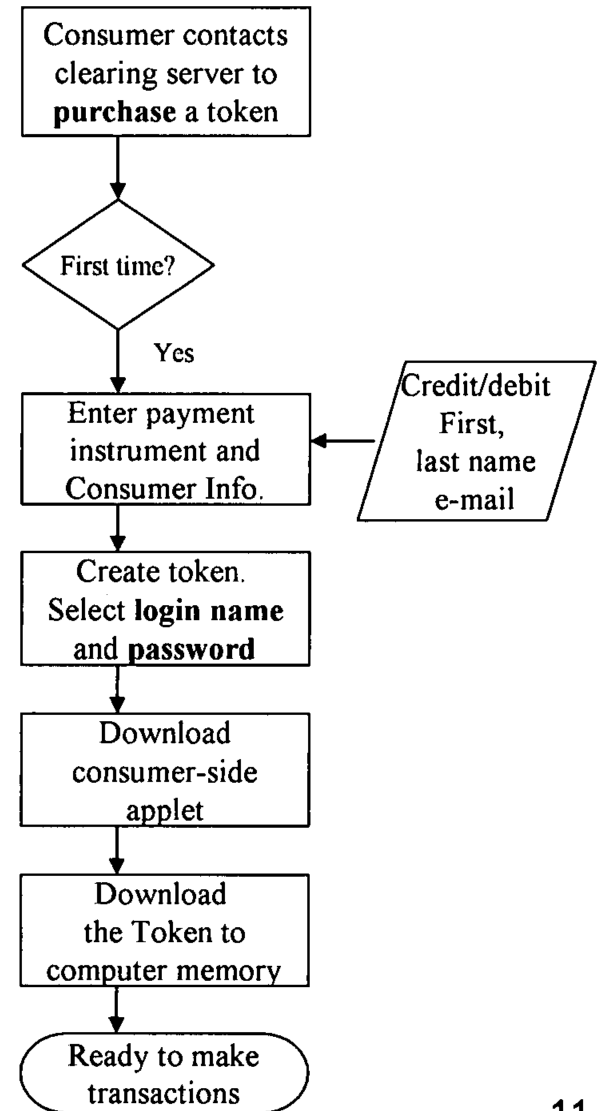


Buying a Token: Technical

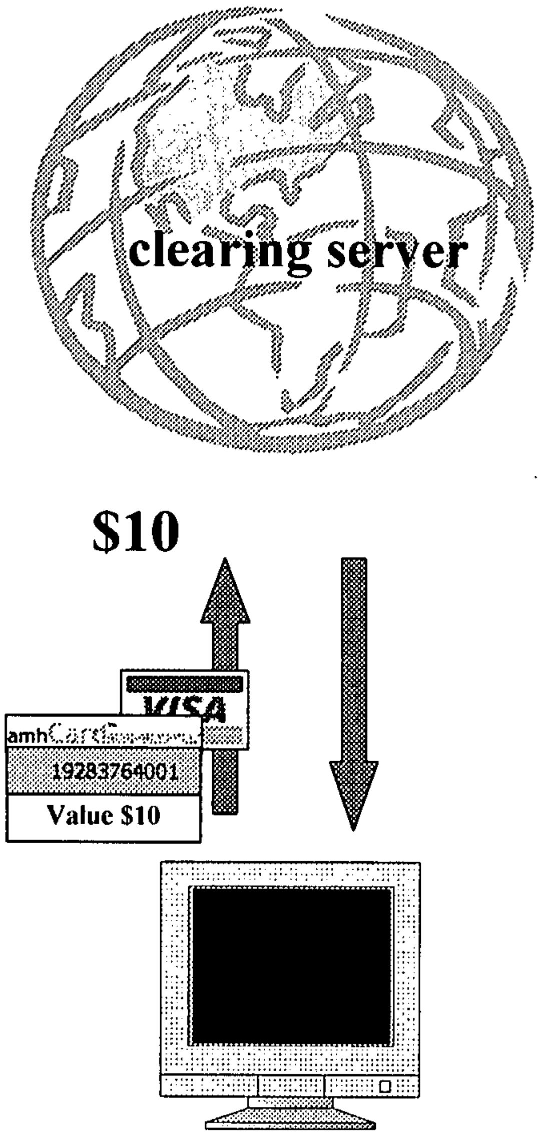


Payment Instrument Verification





Prepaid Token Card



- The Prepaid Token Card is like a phone card. It represents a pre-created Token in clearing server's database.
- Entering the pin number on the Prepaid Token Card downloads the pre-created Token to the consumer.
- Even the clearing server does not know the identity of the consumer. It may be purchased offline by cash, check or credit card.

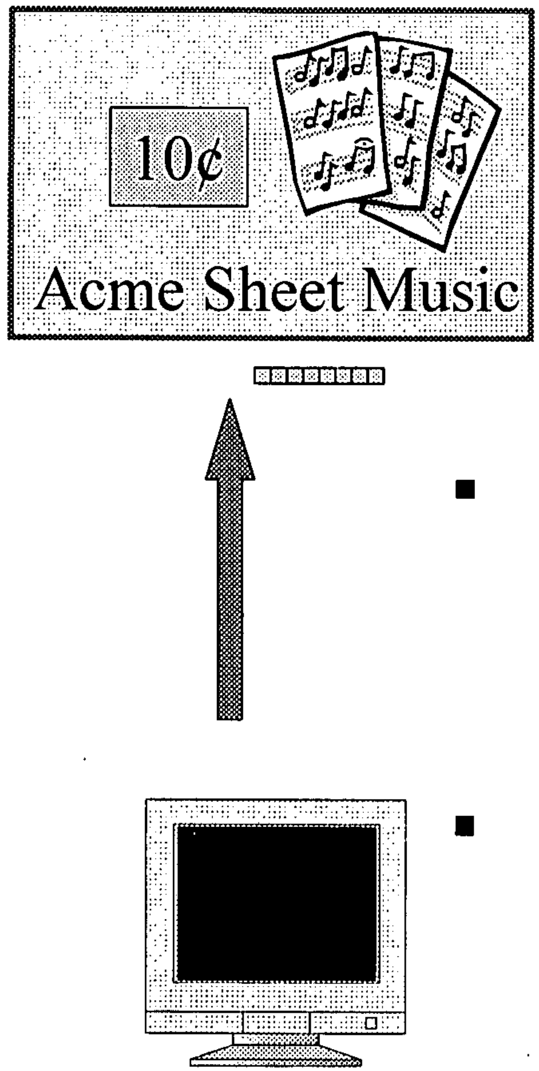
The Token

The Token is like a volatile cookie with logical fields some of which are encrypted. All fields are necessary but the most important fields are:

1. Encrypted Time and Date.
2. Encrypted Token amount
3. Encrypted Hash Value of Token
4. Unique Random Number
5. Last Merchant URL

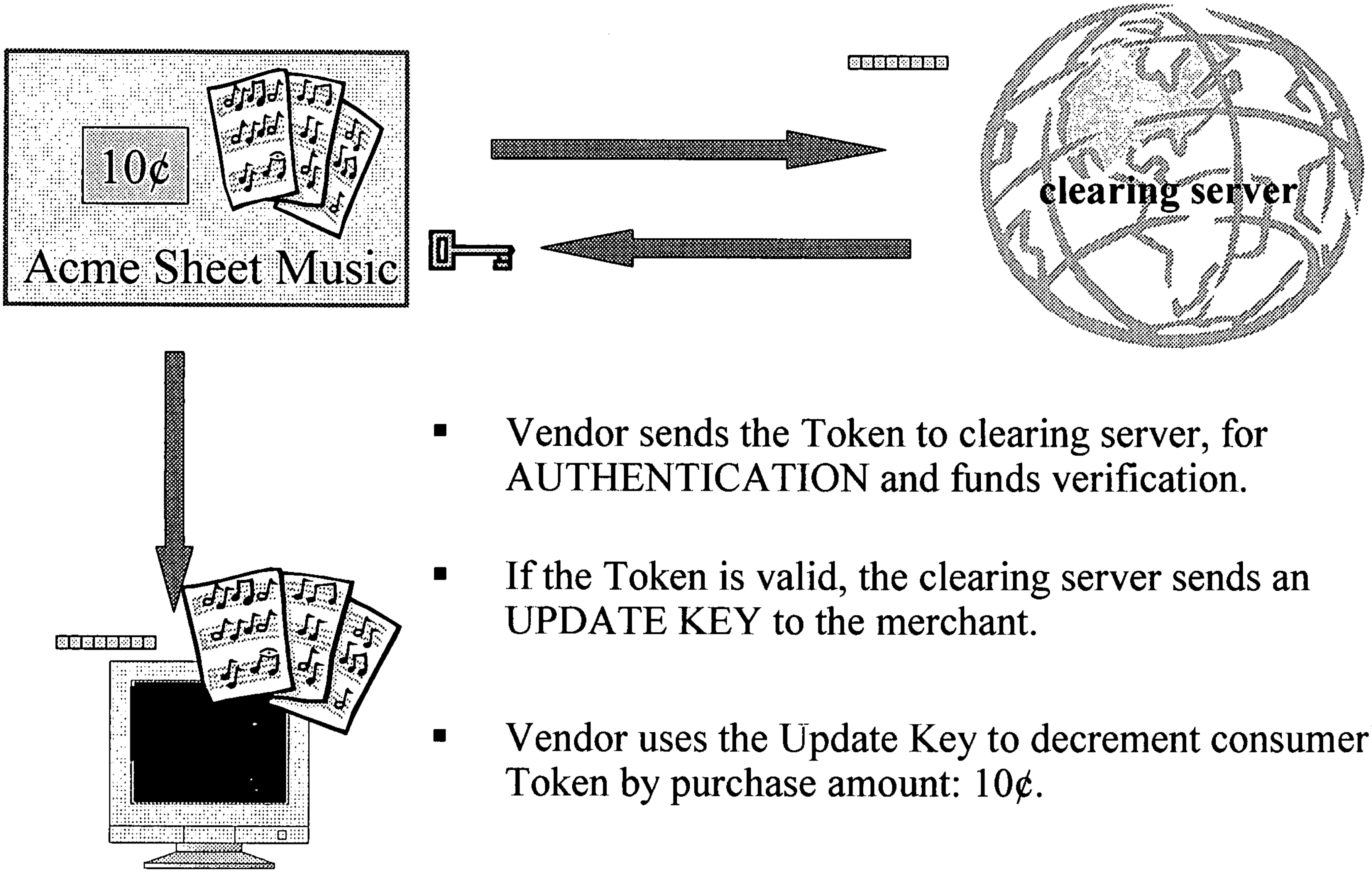
Consumer ID	Database Alias	Encrypted Consumer ID	Encrypted Time and Date (GMT)	Token Amount	Encryption Key Code	Encrypted Token Amount	Merchant ID (Merchant of Last Purchase)	Encrypted Hash value of Token	Encrypted Clearing Server Signature	Unique Random Number	Last Merchant URL

Making a Purchase

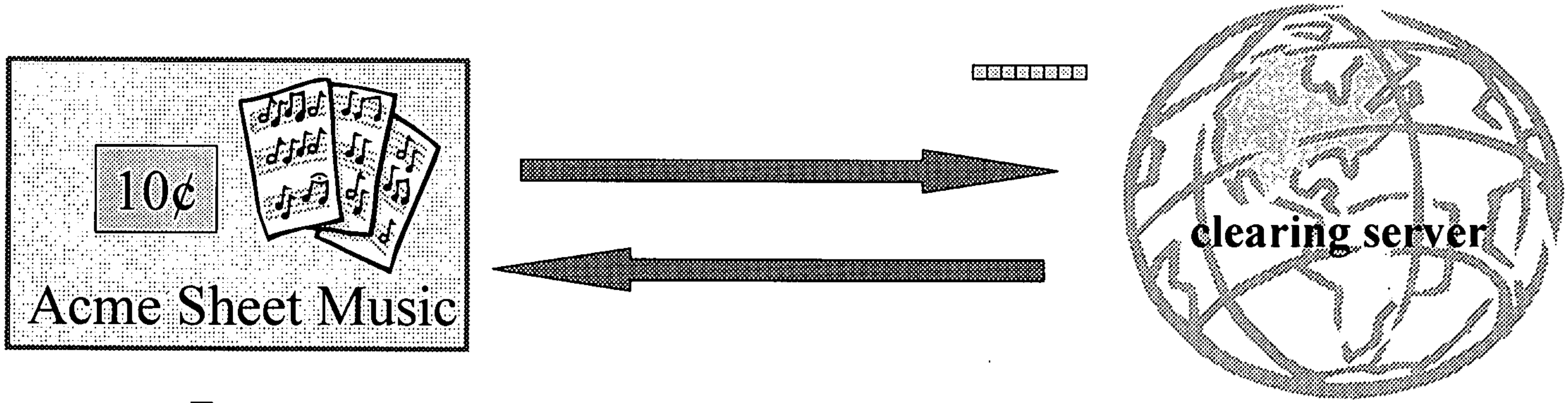


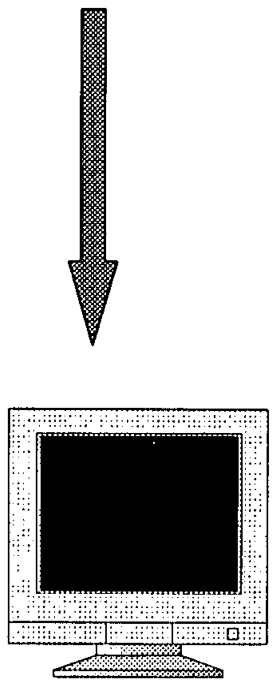
- Using an Internet browser, Consumer visits Sheet Music Vendor website and purchases sheet music by clicking 10¢ icon.

- This action tenders consumer's Token to Vendor as one would give cash, to make the payment.

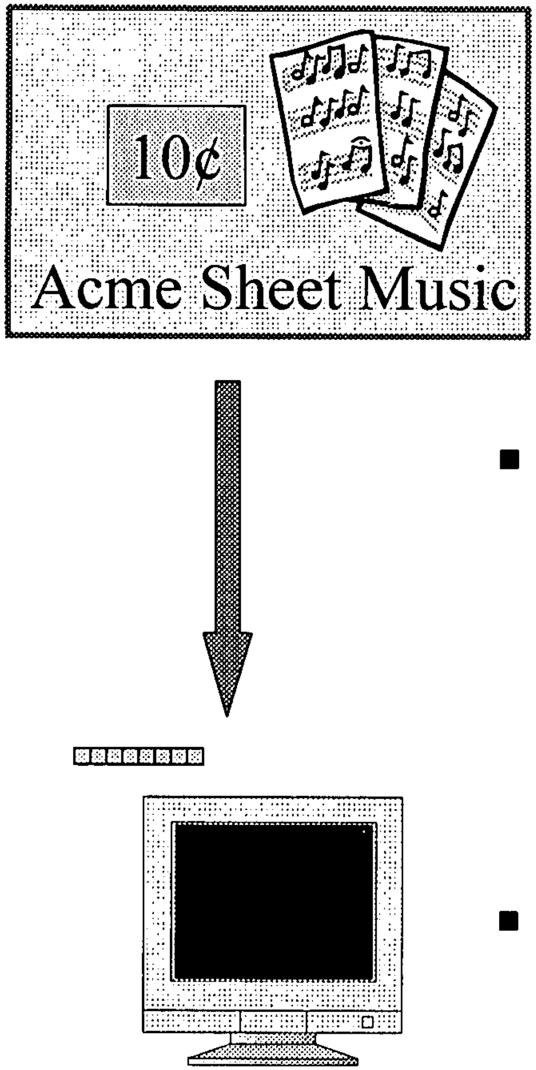


- Vendor sends the Token to clearing server, for AUTHENTICATION and funds verification.
- If the Token is valid, the clearing server sends an UPDATE KEY to the merchant.
- Vendor uses the Update Key to decrement consumer's Token by purchase amount: 10¢.
- Purchased product and the Token are then downloaded to the consumer.





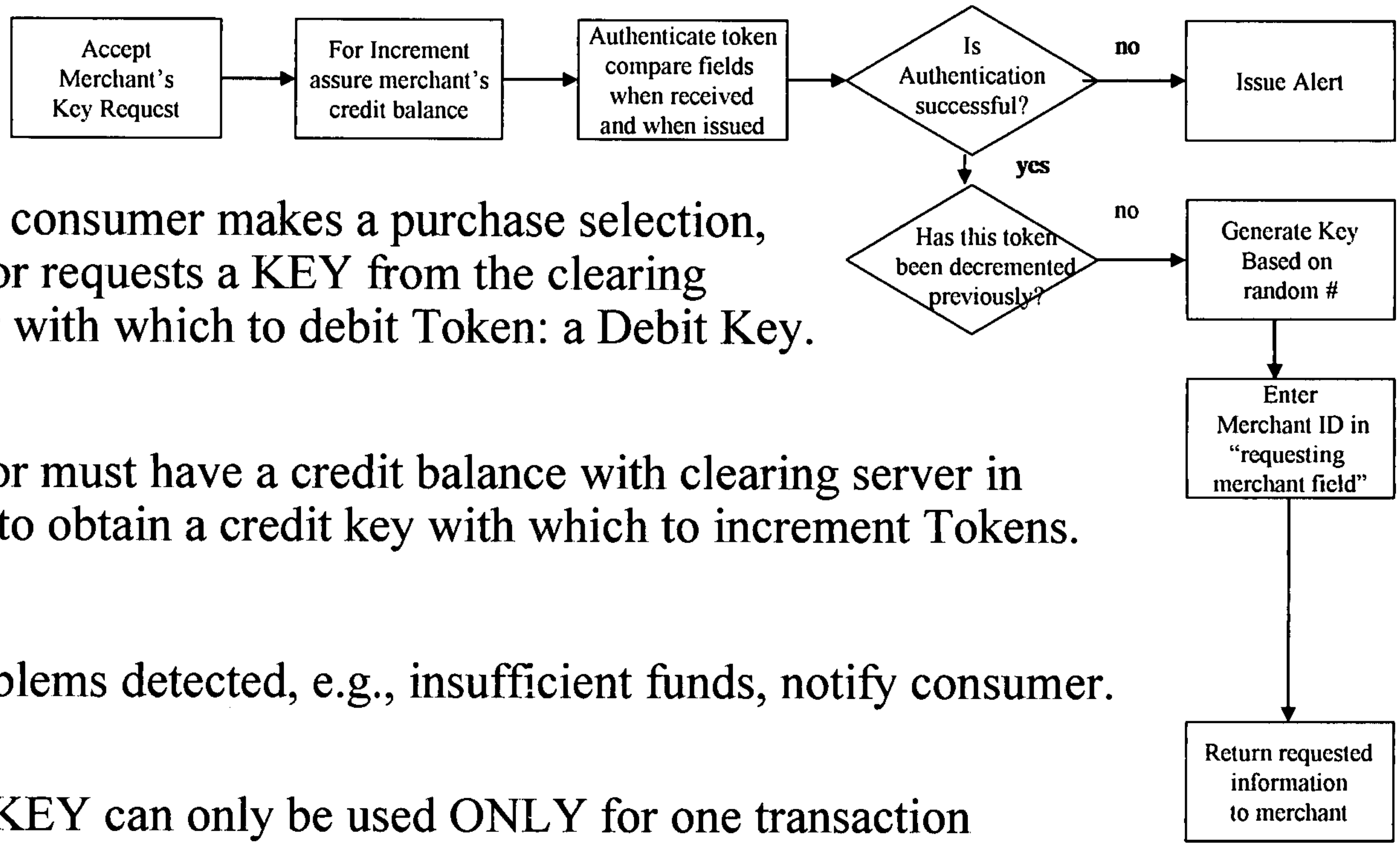
- If AUTHENTICATION fails
- If the token is NOT valid, or funds are insufficient, merchant is notified.
- Merchant informs the consumer



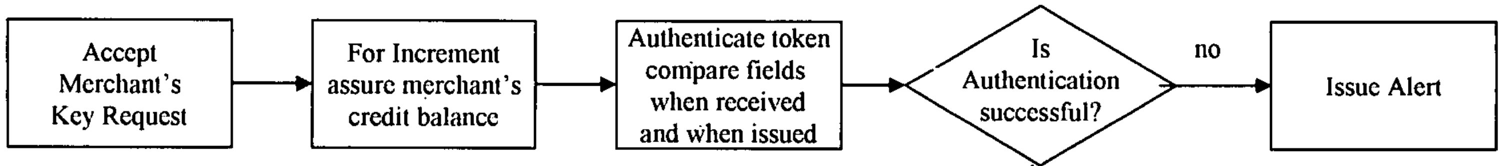
- If an error is made, for example, the product was never delivered. Consumer is credited Vendor can use the Update Key to increment consumer's token. In this example 10¢, is returned to Consumer.

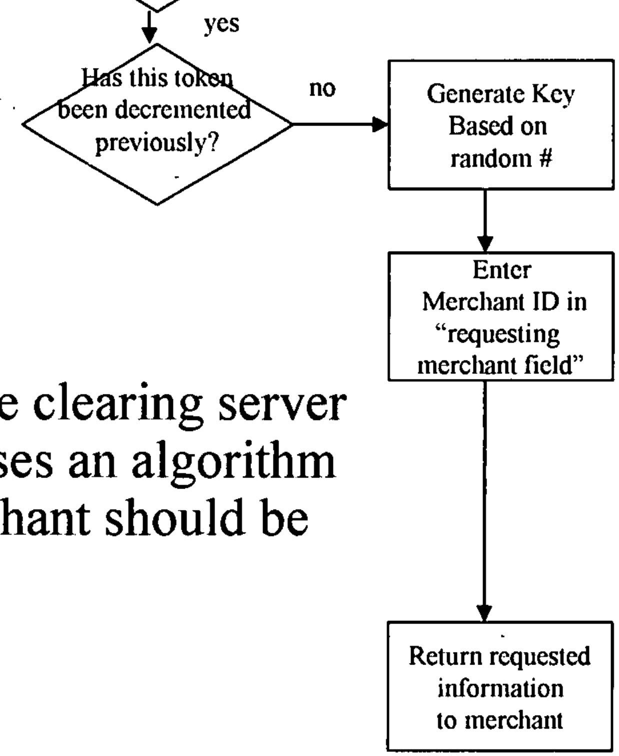
- The Token adjustment is displayed on the Consumer's computing device.

Token Authentication



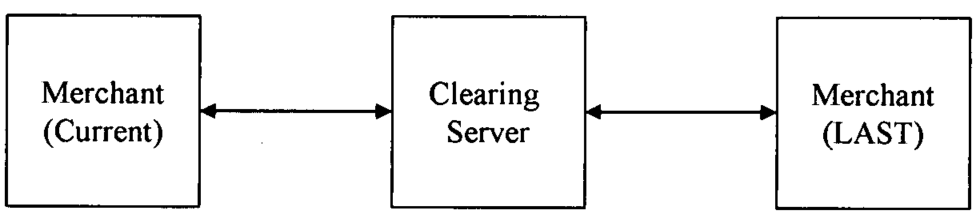
- When consumer makes a purchase selection, Vendor requests a KEY from the clearing server with which to debit Token: a Debit Key.
- Vendor must have a credit balance with clearing server in order to obtain a credit key with which to increment Tokens.
- If problems detected, e.g., insufficient funds, notify consumer.
- Each KEY can only be used ONLY for one transaction session: A session refers to transactions between vendor and buyer, without intervening transactions with another vendor and whose duration does not exceed the life of the Token.

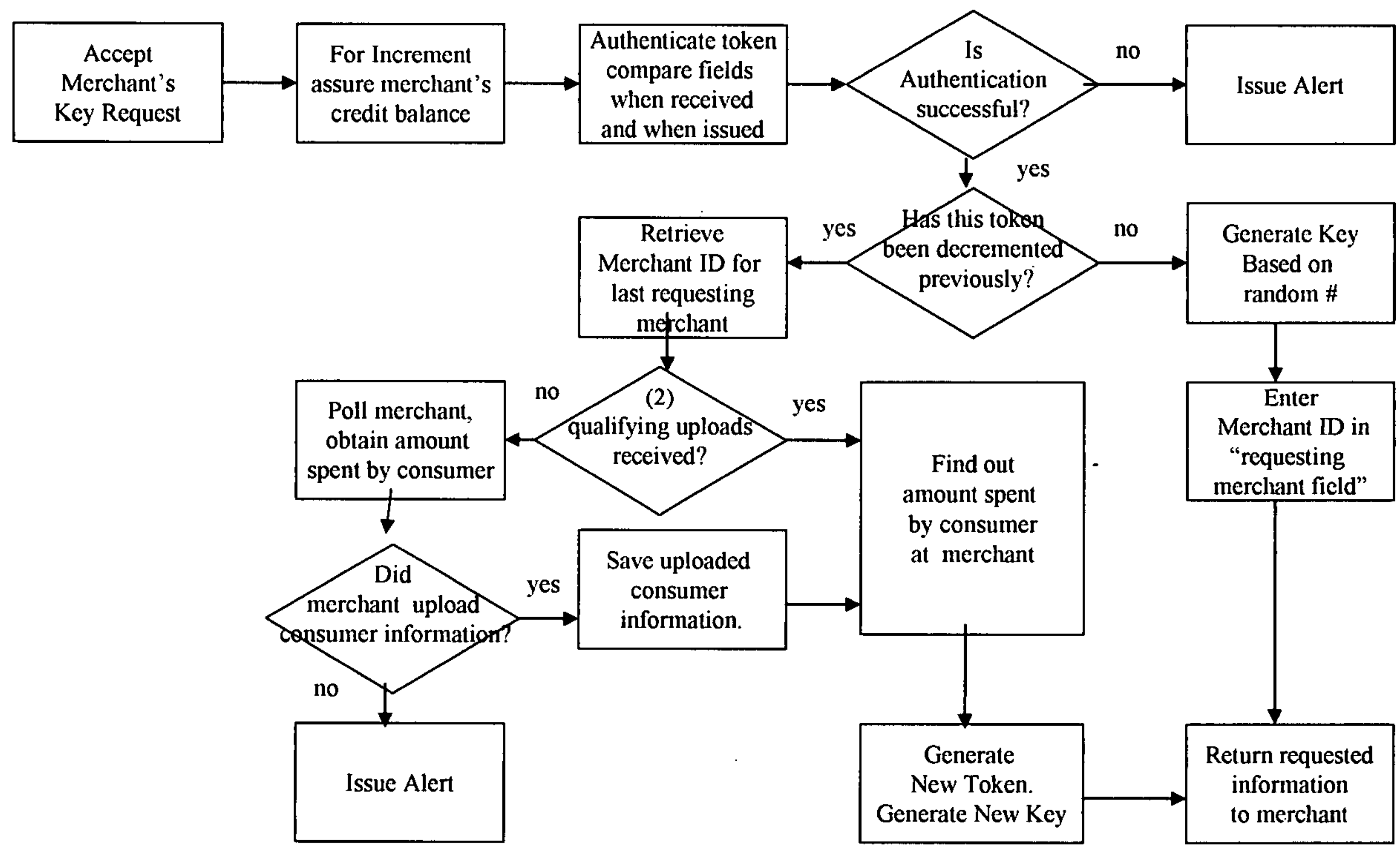




- If the clearing server determines that the Token has been previously used, it polls the merchant with whom the consumer transacted **LAST**.

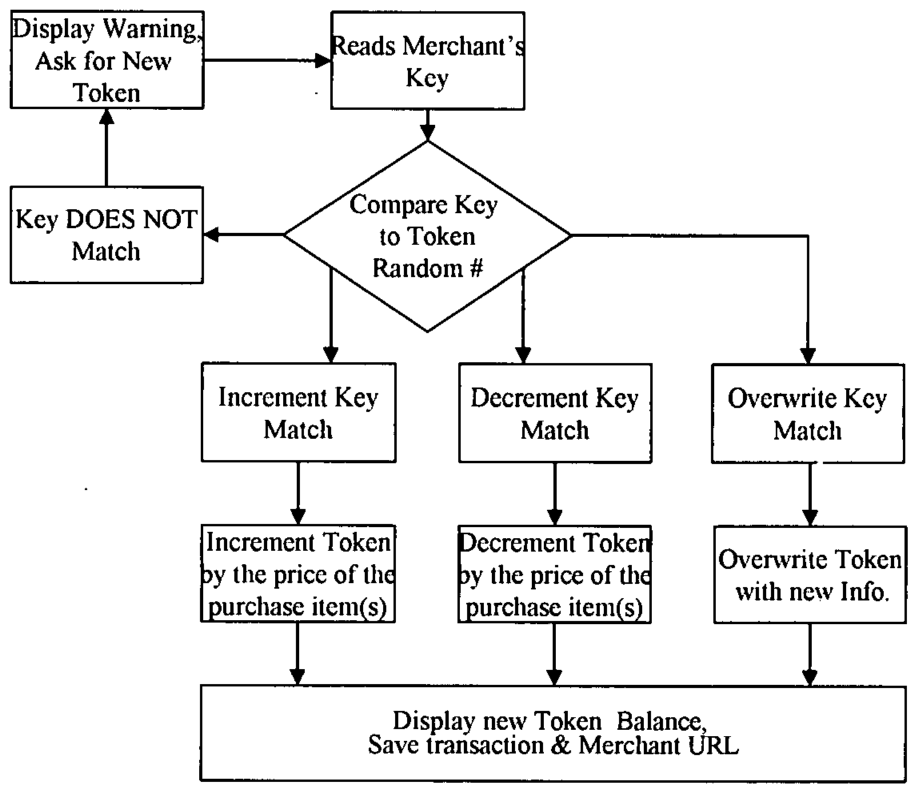
- Merchants upload transaction information to the clearing server at regular time intervals. The clearing server uses an algorithm to determine if an upload is current or if a merchant should be polled for an upload.




Accept Merchant's Key Request
For Increment assure merchant's credit balance
Authenticate token compare fields when received and when issued
Is Authentication successful?
no
Issue Alert
yes
Has this token been decremented previously?
no
Generate Key Based on random #
yes
Retrieve Merchant ID for last requesting merchant
Enter Merchant ID in "requesting merchant field"
(2) qualifying uploads received?
no
yes
Poll merchant, obtain amount spent by consumer
Find out amount spent by consumer at merchant
Did merchant upload consumer information?
yes
Save uploaded consumer information.
no
Issue Alert
Generate New Token. Generate New Key
Return requested information to merchant

Update Key Generation

Update key is used to enable modification, e.g., increment or decrement, of the Token. This is another way of giving change.



A. Consumer makes a purchases at Vendor 1 and then at Vendor 2.

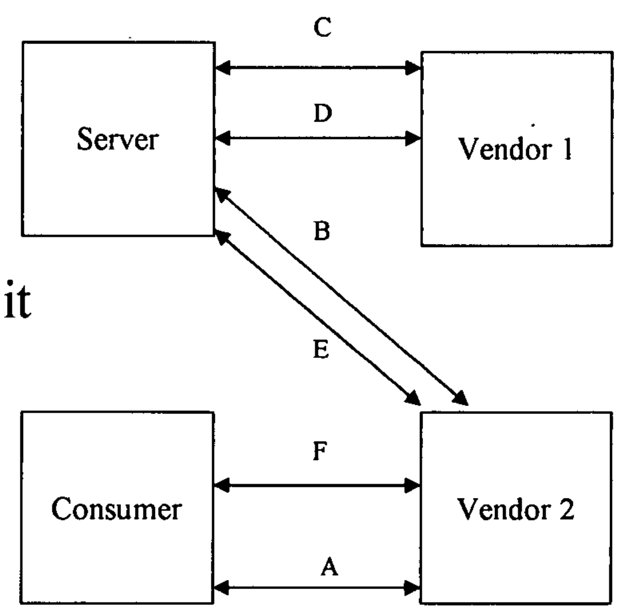
B. Vendor 2 requests from the Server a KEY to charge consumer's Token. <u>The Server records and updates in the Token the new vendor's ID</u>.

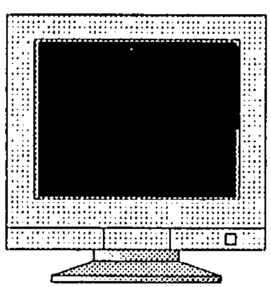
C. The Server determines that Vendor 1 has not uploaded transactional records of purchases made by the consumer. Server requests records upload.

D. Records are uploaded thereby allowing the Server to assure Token value.

E. Server creates a unique session Key and makes it available to Vendor 2.

E. Vendor 2 releases goods purchased by the Consumer.





TheToken is Volatile

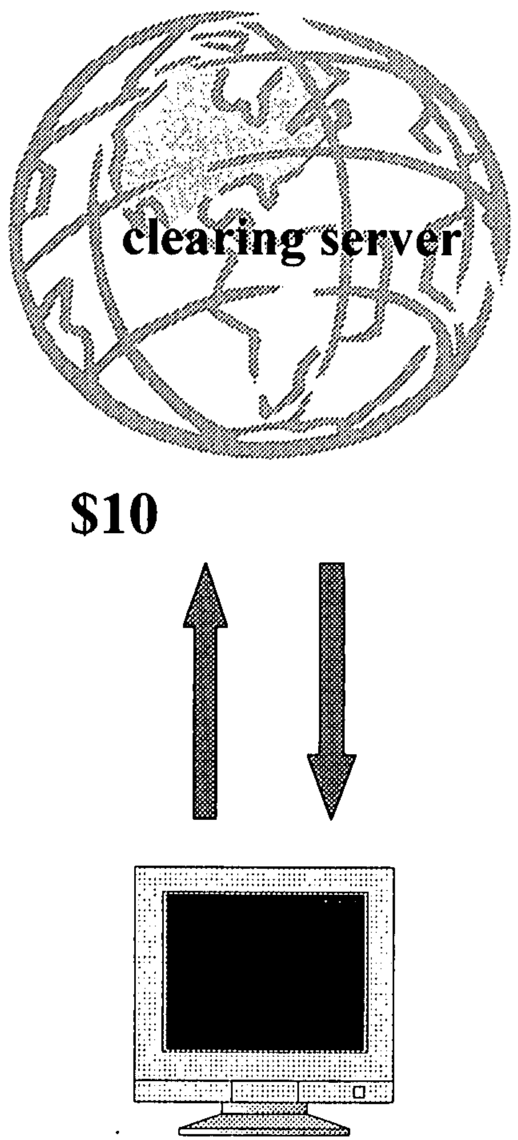
In situations where

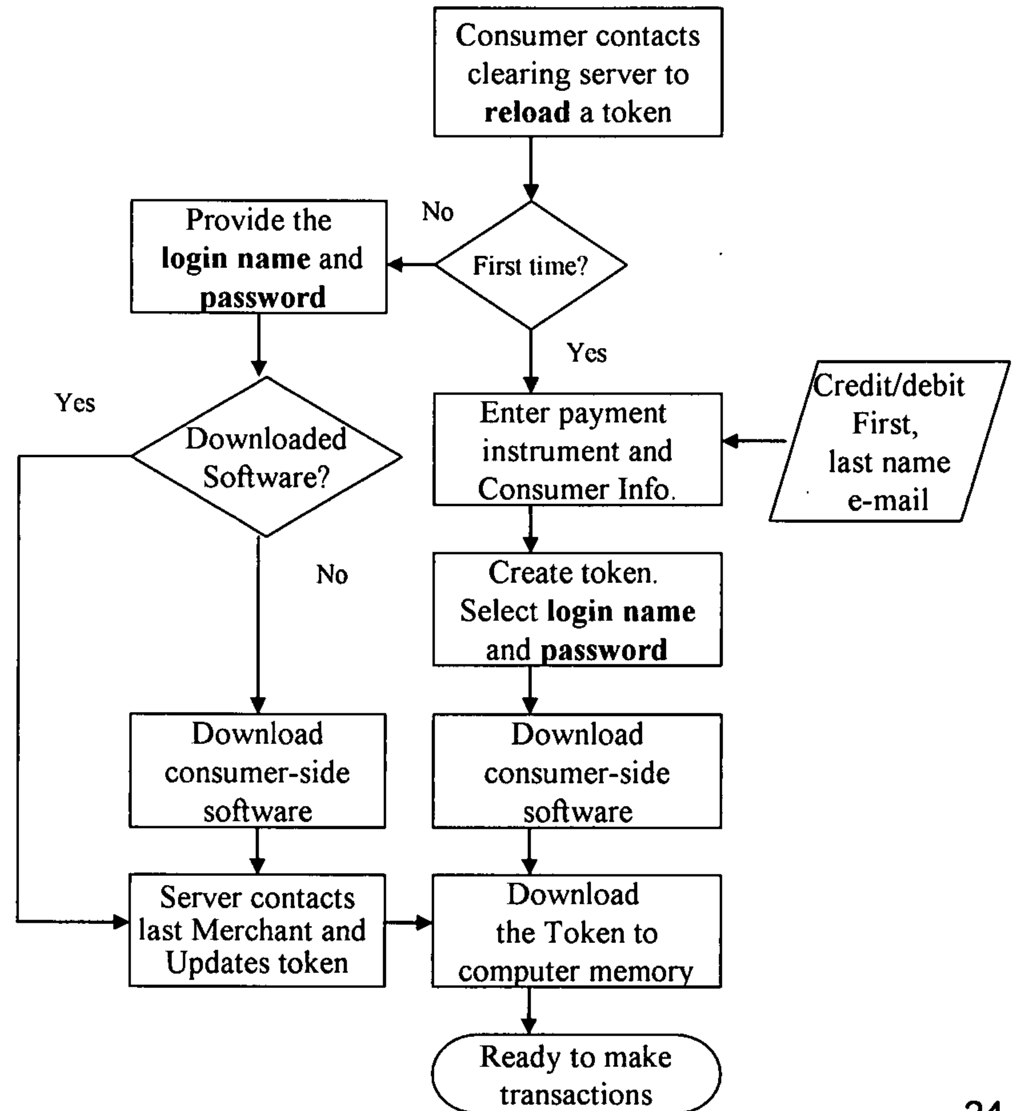
- Consumer decides to stop using the Token;
- Consumer's computer is turned off or loses power; or
- suspicious or unauthorized use of the Token activity is detected; or
- the 2 hour life of the Token is exceeded;

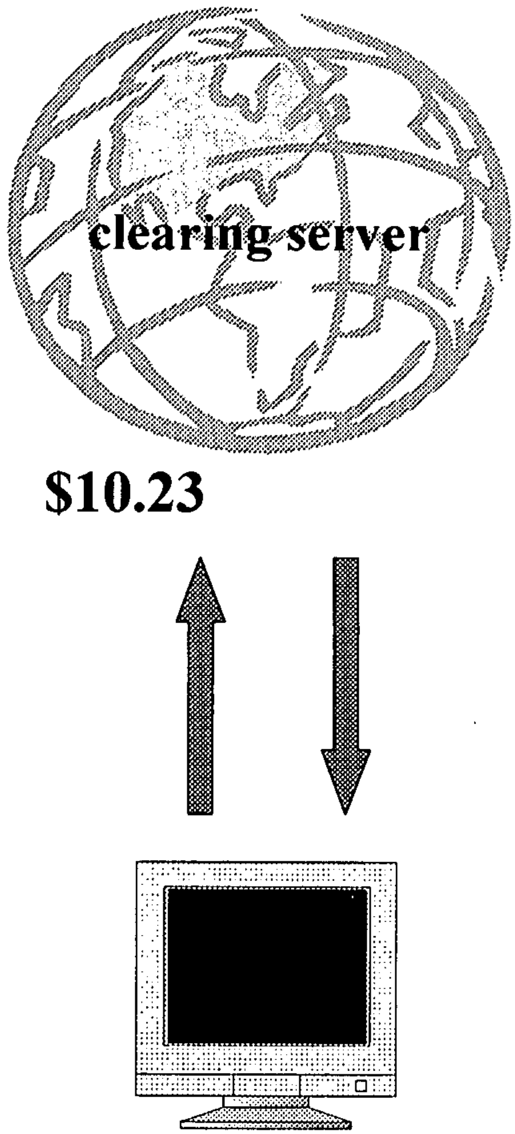
the Token vanishes.

- Consumer simply returns to the clearing server or designated site to reload Token.

Token reload



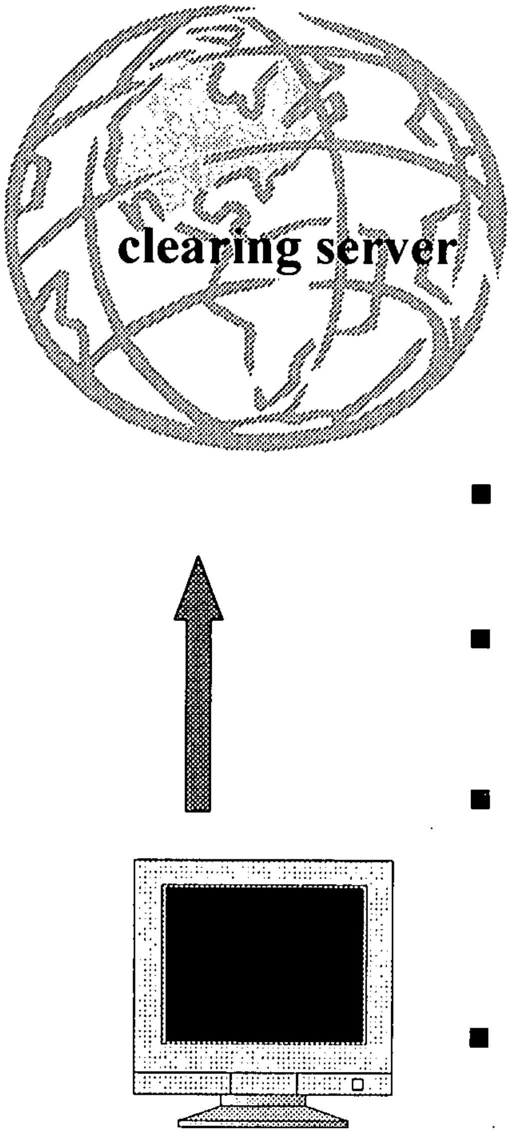




Token Refill

Consumer can replenish the Token at any time by:

- Logging into clearing server's web site; entering his login name and password.
- Requesting funds be transferred to his Token from his credit card or checking account.



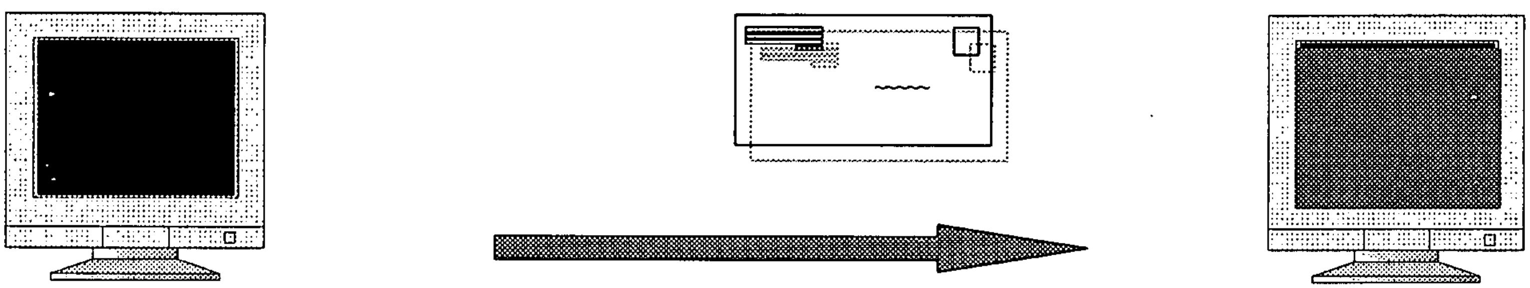
Money Transfer

Tokens can be used to perform money transfers. Transferor only needs to provide

- information with which to identify the transferee;
- the amount to be transferred, e.g. $100;
- clearing server debits Transferor's Token by the transfer amount, in this example $100.
- and generates a unique transfer number.

Transferor then forwards to the transferee

- the selected identifying information, and
- the generated transfer number.



The Transferee will then:

- Connect to clearing server web site; and
- provide identifying information and the transfer number.

- A new Token is created or an existing Token credited for this amount.